THE APPLICATION OF A KNOWLEDGE BASED SYSTEM TO MICRO-ELECTRODE GUIDED NEUROSURGERY

A Dissertation
Presented to
The Academic Faculty

By

Linda Rosemary Harley

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science in Civil and Environmental Engineering

Georgia Institute of Technology
February, 2004

THE APPLICATION OF A KNOWLEDGE BASED SYSTEM TO

MICRO-ELECTRODE GUIDED NEUROSURGERY

Approved by:

Dr. Nelson Baker, Advisor

Dr. Alexander M. Puzrin

Dr. Michael Hunter

Date Approved:

February 4, 2004

Dedicated to

My parents, Ron and June Harley for their love and support

ACKNOLWEDGEMENT

First of all I would like to thank my parents for always being there and believing in me. Both have given sacrificially so that I could follow my dream. Thank you, dad for being a reader on this thesis and for your willingness to always debate the meaning of life issues. My mother, who has served both my father and I out of love. No greater gift has she given me, than by taking care of us both.

To my advisor Dr. Nelson Baker who has given of his time and wisdom, thank you for being there whenever I needed to discuss some of the finer details of this work. Your positive outlook on life, no matter the circumstances, always made things seem better.

To Dr. Klause Mewes for allowing me to be part of this project, it has been interesting and challenging. Maribeth Gandy for assisting do the code interactions, Vince Gibson for helping with the three dimensional modeling tool, Dinal Anderson for his help with the digital signal processor and John Peifer for brining together this team. To NIH (8R01EB002090-04) for funding this research project.

To the rest of my family and friends who have graciously stood by me and supported me to pursue my dreams, thank you.

Most of all I would like to thank the Lord Jesus Christ for giving me this opportunity to serve Him. I dedicate this thesis to Him, for without His guidance and love none of this would ever have been possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

3D-model	Three dimensional model of brain
CBR	Case Based Reasoning
DBS	Deep Brain Stimulation
DSP	Digital Signal Processing
EMG's	Electromyograms
GP	General Practitioner
GPe	Globus Pallidus Exterior
GPi	Globus Pallidus Interior
IC	Internal Capsule
KA	Knowledge Acquisition
KBS	Knowledge Based System
La	Lamina Anterior
Le	Lamina Exterior
Li	Lamina Interior
MM	Modified Mercalli
MRI	Magnetic Resonance Images
OT	Optic Track
PD	Parkinson's Disease
ST	Striatum
UI	User Interface

SUMMARY

Parkinson's Disease (PD) is caused by a depletion of dopamine in the Basal Ganglia region of the brain. PD is most commonly treated by taking L-dopa medication that restores the dopamine levels. Micro-electrode guided neurosurgery can also be used for treating PD in severe cases or when medication does not work. These surgical procedures include a Pallidotomy or a Deep Brain Stimulus (DBS). The overall execution of these two procedures are the same, however the final outcome is different. During a Pallidotomy a lesion is made in the Basal Ganglia, while in a DBS a lead is implanted to stimulate the neuronal cells in the Basal Ganglia. The purpose of the Onetrack system is to assist the surgical team in determining the optimal location of the lesion or DBS lead.

The Onetrack system is composed of several components, including a three dimensional model of the brain (3D model), a Digital Signal Processor (DSP) and a Knowledge Based System (KBS). An MRI is taken of the patient's brain to determine where the first track should be placed. A track is when a probe is inserted into the patient's brain for the purpose of determining the geometrical orientation of anatomical structures. The 3D model takes the information available from the MRI and anatomical information from a brain atlas and generates a computer simulated three dimensional model of the brain. This is used to determine the relative depth values at which the track should find certain anatomical structures. The DSP is used during the track execution to analyze the signal produced by the track as it passes through neurons in the brain. The DSP is capable of assigning specific anatomical structures and neuron cell classifications depending on the signal. The KBS takes the information available from the 3D model,

the DSP and inputs from the surgical team to determine the final anatomical layer that the track is in at a given depth and point in time. Once a track is completed the surgical team looks at the results and determines where the next track should be placed. When the surgical team is confident in the geometrical orientation of the anatomical structures, in specific the Basal Ganglia, a final track is made. During this final track a lesion is made in the case of a Pallidotomy or a lead is implanted in the case of a DBS.

This thesis focuses on the design of the KBS and its implementation into the Onetrack system. In order to design a KBS, it is necessary to gather all the relevant knowledge that should be encoded into the KBS. Therefore, a process called Knowledge Acquisition (KA) is done. KA involves reviewing written material on the subject as well as interviewing experts. The gathered knowledge is examined and important knowledge statements are extracted for encoding into the KBS. The KBS for the Onetrack System consists of two loops, and outer and an inner. The outer loop involves activities that the surgical team does before and after track execution. The inner loop involves all of the activities that take place during track execution. The final result from the KBS is a decision on the anatomical layer and neuronal cell type for the track at a given depth and point in time.

The KBS is verified in two ways. First, the knowledge incorporated in the KBS is tested to the knowledge that is gathered. The KBS for the Onetrack system closely resembles the main ideas and concepts of the knowledge gathered through KA. The KBS is also tested in its programming environment using two actual patient cases. The results from the KBS are good and therefore the KBS can be incorporated into the entire Onetrack system.

By adding these new technologies to the existing surgical procedure, the overall time of the procedure and chances of infection are reduced. This improves the efficiency of the procedure and makes it more desirable for a patient to undergo.

CHAPTER 1

INTRODUCTION

1.1 Introduction

Approximately 1.5 million people in the United States suffer from Parkinson's Disease (PD) [1.] PD is a motor disorder and can be treated by taking medication or by undergoing various forms of brain surgery. This thesis describes the development of a new micro-electrode guided neurosurgical tool that aids the neurologist during a Pallidotomy, a specific surgery for PD. This tool is called the Onetrack system and it aims to reduce the uncertainty of the operation and the surgery time.

1.2 Problem Description

PD is a disease that affects the motor capability of the patient. In some cases the patients are even incapable of physically taking care of themselves. Routine daily tasks such as brushing of teeth or eating become impossible. PD is the degenerative disease of the Globus Pallidus Interior (GPi) in the brain. The GPi controls the motor sensory movements of a person, and a dysfunctional GPi can therefore cause much discomfort. For example, PD patients might have difficulty moving an arm or may have a constant twitching in their hand. PD is most commonly treated with L-dopa medication which is a form of dopamine that causes the GPi to function normally. However, this medication is not successful for all patients. Moreover, the affects of L-dopa wear off after some time, and an alternative solution is therefore needed. Two alternatives are available, namely a Pallidotomy procedure or a Deep Brain Stimulation (DBS) implantation procedure. Each of these procedures typically lasts for 7-12 hours, under local anesthetic. The patient is awake during the entire time and has to respond to requests from the neurologist, for

example to move a particular limb. During the procedure a glass probe moves through the rear of the skull into the brain along a track. As the probe moves, information is gathered about where the different regions of the brain start and end. These regions differ from patient to patient. Obviously it is uncomfortable for the patient to remain in one position for the duration of the procedure, but it is essential for the head to be clamped in a fixed position. The longer the operation takes, the higher the risk of infection. This is also a risky procedure because of the uncertainty involved. A track is defined as the trajectory from the point where the probe enters the brain to the point that it is retracted from the brain. The neurologist has to try several tracks in order to determine the anatomy of the patient's brain and decide on the best location for the probe before an electric discharge is created at the end of the probe tip to make a lesion. The lesion destroys the GPi. This is an irreversible process, and if the lesion is not placed in the correct location it can lead to blindness because the Optic Track is located close to the GPi.

During a DBS, a lead is implanted into the patient's brain instead of creating a lesion. The signal that the lead emits in the brain can be adjusted after the operation in order to obtain the most desired affect. The accuracy and speed of a Pallidotomy or DBS procedure are improved with better MRI scans and by the addition of the Onetrack system. The Onetrack system also reduces the risk of infection, the risk of blindness and improves the placement of the Pallidotomy lesion or the DBS lead.

The Onetrack system consists of various components and contains new technologies that are currently not available to the neurologist. The technological components of the Onetrack system consist of a three dimensional model of the brain, a digital signal

processor, improved MRI scans of the brain and a Knowledge Based System. An MRI scan is taken of the patient's brain for the purpose of determining the trajectory of the first track. Once the MRI is performed, a three dimensional model of the brain is created by the computer. This gives the neurologist an idea of where the probe will move with respect to the geometry of the anatomical structures in the brain and helps to plan a path trajectory. A path trajectory is a list of probe depth values that correspond to certain anatomical layers. During track execution the digital signal processor analyzes the signal that is produced by the probe, and produces a list of probable cell types. Without this new tool the neurologist has to distinguish certain cells and layers by listening to the signal of the probe. The digital signal processor therefore aids in the identification of cell types which is used to verify that the probe is moving along the desired predetermined trajectory. The Knowledge Based System uses information from the three-dimensional model, the digital signal processor and the neurologist to identify the location of the probe. The Knowledge Based System therefore identifies the final cell type and the layer in which the probe tip is located at a certain depth at a given point in time. After the completion of a track, the Knowledge Based System suggests where the next track should be placed. When all the necessary tracks have been executed and the neurologist has a good indication of the geometry of the brain, a final track is made in order to make the lesion or place the DBS lead. The primary function of the Onetrack system is to give guidance to the neurologist throughout the procedure.

1.3 Hypothesis

This thesis examines the development of one of the primary components in the Onetrack system, namely the Knowledge Based System which is the artificial intelligence for the Onetrack system. The Knowledge Based System continually analyzes the data from the

other components of the Onetrack system, provides a decision on the probe tip location within the brain and aids the neurologist with the procedure. These decisions are based on knowledge gathered from experts in the field of neuroscience, and the data from various measurements taken from moment to moment as the procedure is progressing. The central premise of this research is that a knowledge based system can be developed to assist in guiding the surgical team with a Pallidotomy or DBS procedure.

1.4 Thesis Organization

To understand the background of the procedure, a more detailed description of PD and how it is treated is presented in Chapter 2, which also introduces some of the basic concepts of the Pallidotomy and DBS procedures. Chapter 3 presents the components of the Onetrack system. One of these components represents the expert's knowledge, and several alternate solutions for representing this knowledge are examined in Chapter 4; one of these solutions is the Knowledge Based System (KBS) chosen. A description of the design process of a KBS is given in Chapter 5. One of the phases in the design process is knowledge acquisition and Chapter 6 explains how the knowledge is gathered and analyzed. The knowledge gathered is divided into two primary groups, namely the process knowledge and the domain knowledge, and the implementation of this knowledge is described in Chapter 7. Chapter 8 explains the communication protocol that is designed to create the interaction between the Knowledge Based System and the other Onetrack system components. The knowledge contained within the Knowledge Based System is tested and validated, and the methodology and results for these tests are found in Chapter 9. Lastly, other possible applications to Civil Engineering are examined and probable future work is discussed in Chapter 10.

1.5 Summary

A Pallidotomy procedure or a DBS procedure is performed on patients who suffer from PD. The Onetrack system aims to improve this procedure by providing the neurologist with new tools such as a three dimensional model of the brain, digital signal processing of the neural discharge, and a Knowledge Based System as a guidance system. The Knowledge Based System aids the neurologist by suggesting the possible location of the track within the three dimensional space of the brain. This thesis describes the development of this Knowledge Based System and the author is referred to as the knowledge engineer.

CHAPTER 2

REVIEW ON PARKINSON'S DISEASE

2.1 Introduction

The Onetrack system is a computer aided software and hardware package that assists a neurologist in performing brain surgery on people who suffer from Parkinson's Disease (PD). The intensity of Parkinson's symptoms varies from patient to patient and are most commonly treated by medication. However, in severe cases it is necessary to perform brain surgery to alleviate the symptoms. The two brain surgeries that the Onetrack system supports are Pallidotomy and Deep Brain Stimulus (DBS), both are complex procedures that can last from 7 to 12 hours. This chapter will discuss the history, causes, and symptoms of PD, followed by medical treatments with a concentration on the Pallidotomy and DBS surgical procedures.

2.2 History of Parkinson's Disease

PD, first identified by James Parkinson a 19th century British physician, "is a complex, common, chronic brain disorder that results primarily from the progressive death of a specific group of nerve cells in a layer of a region of the substantia nigra in the midbrain" [2]. The death of the nerve cells is caused by the loss of a chemical substance called dopamine. Two solutions for alleviating the symptoms of PD have developed over the years; one is an invasive brain surgery and the other a treatment of L-dopa medication. "The first experimental surgery for Parkinson's Disease was done by Myers in 1939 with lesions in the basal ganglia" [3]. Over the years the placement of these lesions has improved with the addition of new brain imaging and signal processing technologies. In the late 1960's L-dopa therapy became readily available and almost completely

replaced the brain surgery. However, it did not succeed due to severe side affects of taking L-dopa and poor long term results [3]. Today the brain surgery is performed on people who suffer from severe cases of PD and cannot take the medication.

2.3 The Cause of Parkinson's Disease

In 1960, Ehringer and Hornykiewicz [4] demonstrated that PD is caused by the depletion of cerebral dopamine (DA) in the brain. The brain consists of several layers, with those of interest to this project described below.

The brain is structured and the order in which the layers occur is consistent between different people. Figure 2.1. gives a visual description of the anatomical layout of these structures. The outer layer is called the Striatum (ST), which is followed by the Globus Pallidus Exterior (GPe). The GPe and the Globus Pallidus Interior (GPi) together are called the Basal Ganglia. The Optic Track (OT), the Internal Capsule (IC) or the Nucleus Basalis are located below the Basal Ganglia. Between these primary layers are the Lamina (secondary) layers, the Lamina Exterior (Le, between Striatum and GPe), the Lamina Interior (Li, between GPe and GPi), and the Lamina Anterior (La, below the GPi).

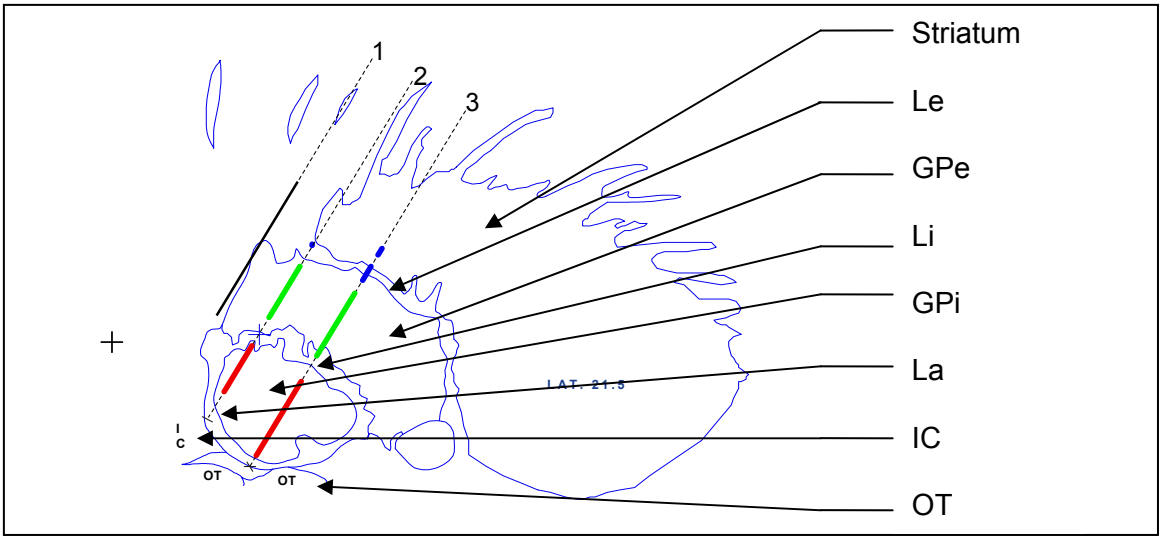


Figure 2.1. Anatomical layout of the brain.

The loss of dopamine primarily occurs in the GPi. PD is alleviated by restoring the dopamine chemical balance in the GPi or by disabling the functions of the neurons in the GPi altogether.

2.4 Symptoms of Parkinson's Disease

Parkinson's Disease is a motor disorder. The major symptoms of PD include tremors in the body extremities, muscular rigidity, and akinesia [5], as described below.

2.4.1 Tremor

A tremor is a rhythmic movement in the extremities, such as in fingers, hands or toes. Tremors have a tendency to intensify whenever the patient is focusing on a specific action. Certain cells in the GPi have a direct correlation with the tremor movements. "Tremor at rest is a cardinal sign of PD. Electromyograms (EMG's) show rhythmic activity alternation in antagonistic muscles at frequencies between 3.5 and 7 Hz." [5, 6]. The inconvenient part of tremors is that the patient cannot exercise any control over the

movement. For example, if the tremor is in the hand, it prohibits the patient from writing, eating and performing other life functioning abilities. In severe cases this tremor acts continuously or near-continuously throughout the day.

2.4.2 Rigidity

Rigidity causes the patient to experience difficulty in the passive movement of limbs. Typically, the slower the movement, the higher the resistance is to move. However, in some patients rigidity is not related to the velocity of the movement, instead the muscle remains rigid once it is stretched. For example, this can be observed in a patient who sits down and then has difficulty getting up because the thigh muscles are stretched out. "In any given patient, the degree of rigidity is not necessarily constant; stiffness is clearly reinforced by contra lateral movement, stress and anxiety" [5]. It is very stressful when one cannot control a part of one's body. The more the patient focuses on this inadequacy, the higher the anxiety becomes and the harder it is to move the limbs.

2.4.3 Akinesia

Of the three symptoms, akinesia is the most disabling and can be characterized as "delayed motor initiative, slow performance of coluntary movements (bradykinesia), and difficulty reaching a target with a single continuous movement (hypokinesia)" [5]. Patients that suffer from severe akinesia are prime candidates for brain surgery, because specific areas of the brain which control certain limbs of the body, can be treated. For example if the patient has difficulty walking because of delayed movement, then the specific area in the GPi responsible for leg movement can be targeted. Akinesia is not that easy to identify in a patient as the symptoms are subtle.

2.5 Medical Treatment for Parkinson's Disease

Currently two main treatments exist to alleviate the symptoms of PD. The first is the administration of L-dopa, an oral medication that affects the chemical balance of dopamine (DA) in the Basal Ganglia and secondly, an invasive procedure that disables certain cell functions in the Basal Ganglia.

By 1967, Catzias et al. [7] had proven the dramatic affect of treating patients with L-dopa. "The basic rationale behind the introduction of L-dopa into antiparkinsonian drug treatment, was that it would act by being converted into DA, thus correcting the striatal transmitter deficit" [8]. However, the effectiveness of L-dopa diminishes when taken over an extended period. In addition, adverse side effects which occur in some patients include nausea, sleepiness, and skin problems. These effects are typically treatable. However, special untreatable problems can arise such as hyper salivation, hyperdrosis, and depression [9]. When L-dopa becomes completely ineffective the only treatment remaining is brain surgery.

There are several types of brain surgery that can be performed to alleviate PD symptoms. These are Thalamic surgery, Subthalamic surgery, Pallidotomy and Deep Brain Stimulus (DBS). The differences between these are the area of the brain that is targeted and the method used to disable the cell functions. This thesis focuses on the Pallidotomy and DBS procedures. Both of these target the Globus Pallidus Interior (GPi). In the Pallidotomy procedure, lesions are made, and in the DBS procedure, an electrical lead is implanted into the brain.

> "The ideal candidate for surgical treatment of Parkinson's disease is a patient who
> (a) has unilateral tremor and little or no rigidity, bradykinesia, mental deterioration, or speech or gait disturbance;
> (b) is unresponsive to, or intolerant of drug treatment; and
> (c) has no general medical risk factors that would increase the risk of surgery" [10].

It is important to note that the right side of the brain controls the left side of the body and the left side of the brain controls the right side of the body. Therefore, if a patient has a tremor in the left hand, then the right GPi is targeted. Typically if the patient has tremor on both sides of the body, two separate procedures are done. The second procedure is sometimes not as affective as the first. If this is the situation then the patient will typically have one Pallidotomy and one DBS done to maximize the affect.

2.6 Pallidotomy versus Deep Brain Stimulus

In both the Pallidotomy and DBS procedure; the objective is to place the lesion or lead in the optimized location by alleviating the PD symptoms most effectively and limiting side affects. The first portion of both procedures is identical in. A probe runs downwards into the skull multiple times to determine the geometrical anatomical constraints of the primary structures in the GPi. Each probe entry is the start of a new track. A single track can take 30 minutes to 2 hours to complete. After the GPi area is mapped, the lesion track or DBS lead implantation is undertaken. Figure 2.2 shows the equipment setup.

Figure 2.2. A photo of a patient in the operation room as the probe enters the brain.

In a Pallidotomy it is crucial that the lesion location is accurate as the lesions is final and cannot be reversed. If the lesion is too low, it can cause a speech impediment or the patient can loose a portion of their eye sight. The lesions are stacked one upon the other along a track to cover the entire GPi area. After a lesion is made in a track, the probe is moved up a few millimeters to make the next lesion. Typically the lesion lead is heated to somewhere in the range of 60 to 80 degrees Fahrenheit. Usually three lesion tracks are required, to cover the entire volume of the GPi, effectively destroying all the neurons in the GPi.

During a DBS, a special lead is permanently placed in the patient's brain, allowing for improved control over the procedure outcome. The strength of the voltage emitted by the DBS lead is controlled by a dial in the operation room. Thus the neurologist is able to adjust the intensity to find the optimal solution to alleviate PD symptoms. After the lead is placed in position, the patient undergoes the rest of the implantation. This involves placing a neurostimulator that regulates the DBS lead, under the skin near the collar bone. One example of the equipment set used to make this possible is called the Activa System, produced by Medtronic. "During the procedure to implant an Activa System, a lead with four 1.5mm electrodes is surgically implanted in the brain and connected by an extension that lies subcutaneously to a neurostimulator implanted near the clavicle. The electrical stimulation can be noninvasively adjusted to meet each patient's needs" [11]. As time passes the intensity of the voltage, produced by the lead, can be adjusted to compensate for the increasing symptoms of PD. If a problem arises requiring equipment replacement, another DBS surgical procedure must be conducted.

The neurologist is the best equipped to assist the patient in selecting either the Pallidotomy or DBS procedure. This decision may depend upon the severity of PD symptoms, medication effectiveness, and procedures previously undergone by the patient.

2.7 Day of the Operation

The following procedure is actually carried out at the Emory University Hospital. A Pallidotomy or DBS procedure can take 7 to 12 hours to complete. Both procedures being with a head frame being screwed into the skull of the patient. The head frame serves as the relative frame of reference for the entire procedure. A Magnetic Resonance Image (MRI) is then taken of the brain. MRI slices are taken approximately every 3 millimeters apart along the parasagittal plane. These image slices are compared to an anatomical model of the brain, such as the Shaltenbrand and Bailey atlas (brain atlas) [12]. Figure 2.3 shows that the parasagittal plane runs parallel to the mid-sagittal plane, which is longitudinal, vertical, dividing the brain into two halves (medial and lateral) [13]. By comparing the patient specific MRI to the general brain atlas the neurologist decides on the optimal position to run the first track.

Unlike normal surgical procedures, the patient is only given a local anesthetic and is not allowed to have general anesthesia. This means the patient is conscious during the entire operation and aware of the surroundings. This is necessary as it is important that the neurologist be able to communicate with the patient during the operation. Once the patient is in position, a craniotomy is preformed (the neurosurgeon drills a hole in the skull). The probe is attached to the head frame. The neurosurgeon adjusts the

trajectory of the probe, as discussed with the neurologist. The probe then enters the brain to start the track.

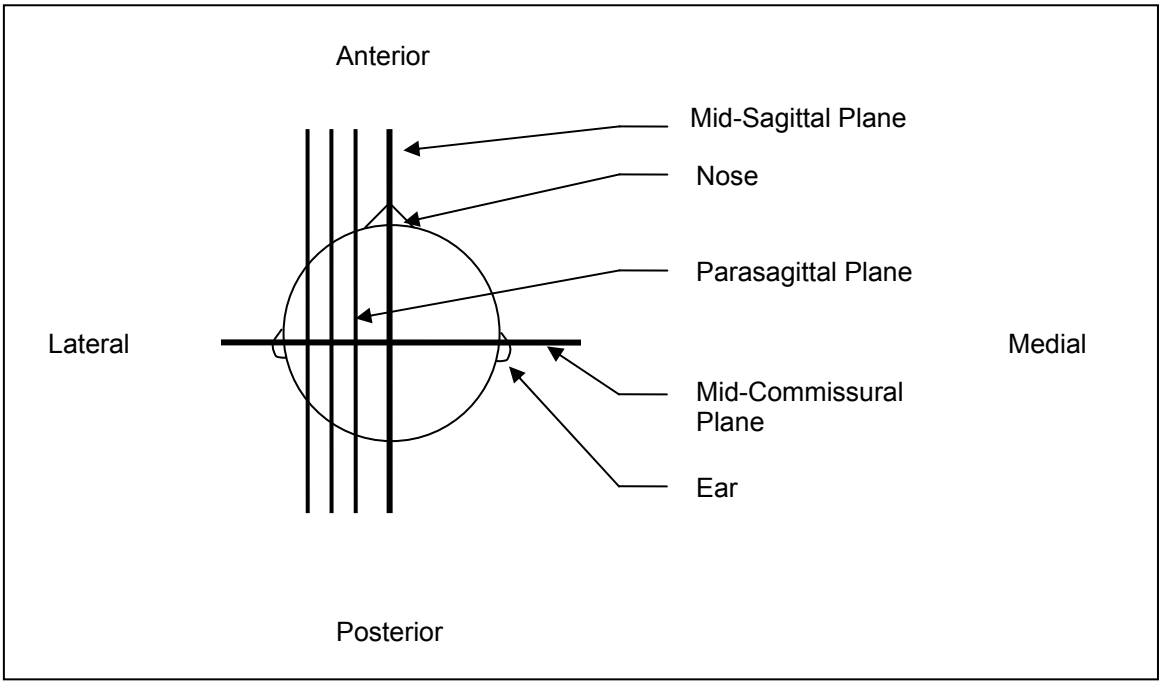


Figure 2.3. The orientation of the parasagittal planes.

The first track tends to take the longest because of the exploration and uncertainty involved. The neuronal cells emit a signal that is captured by the probe. Using the Stealth^{TM} system the neurologist is able to listen to and view the signal. Through experience the neurologist is able to identify each neuronal cell type. Certain cell types correspond directly to a specific layer in the brain. In this manner the neurologist can determine the start and end depth of each primary layer. The primary layers that need to be identified are the Striatum, Globus Pallidus Exterior (GPe) and Globus Pallidus Interior (GPi). Upon starting the track the neurologist predicts at what depth the probe encounters each layer. The neurologist is constantly comparing the predicted results, obtained when determining the trajectory of the probe, to the actual results. These

predictions are based on the MRI and brain atlas. Depending on how close the actual layer depths are to those predicted, is an indication of the track accuracy.

Certain passive or active movements correlate to specific regions of the GPi. Once the neurologist determines the probe is in the GPi, sensory-exploration is done. The patient is awake at this point and may be asked to move an arm, fingers, legs or toes. If the movement corresponds with activity in the probe signal, then the neurologist is assured that the probe is in the GPi.

At the end of the track there is a possibility of finding one of two layers or none (see Fig. 2.1). The Optic Track (OT) is tested by flashing a light into the patient's eye and inquiring whether he/she sees dots. If the response is affirmative then the probe is close to the Optic Track. For the Internal Capsule (IC), macro stimulation is done via the probe. Macro stimulation is when the probe tip increases substantially in the voltage output, so much that it activates certain neuronal cells. If the patient's hand, face, mouth or tip of tongue twitches relative to the on/off flicking of the macro stimulation, then the probe tip is close to the internal capsule. These tests are done to determine where the end of the track is on the anterior, posterior, medial and lateral plane (Figure 2.3.).

After track execution is completed, the graph paper upon which the layer findings are plotted by hand is compared to the MRI and the generic brain atlas. Depending on (1) the accuracy of the first track and (2) the actual information from the track, the neurologist then visually determines where in the brain the track actually ran and where the next track should run. This process is called mapping. A golden rule followed by the

neurologists is that they do not step the probe in two directions at once, but prefer to stay in the same parasagittal plane.

Generally four to six tracks are executed to determine the geometrical anatomical constraints of the GPi relative to the head frame. However, if brain shift occurs, then the track executions start all over because there is no way of knowing how the brain shifted relative to the head frame. A brain shift can occur if there is too much cranial fluid leakage, the patient has a seizure, or the probe stays in the brain too long.

Once the mapping is completed, the lesion track is performed in the case of a Pallidotomy, or a DBS lead is inserted in the case of a DBS procedure. These leads are often thicker than the exploration probe, necessitating the need to limit the number of times that the probe enters the brain. Every time the probe enters, neurons are destroyed.

The “Onetrack” system aims to improve this procedure by introducing new visual and analytical tools to (1) provide a greater degree of certainty for probe placement in the brain; and, (2) to reduce the number of tracks needed to determine location of the GPi.

2.8 Summary

Parkinson’s Disease is caused by the loss of the chemical substance dopamine in the brain, more specifically in the GPi, of the Basal Ganglia. Tremors, rigidity and akinesia are the three major symptoms of PD. Akinesia is the most disabling, and patients with akinesia are commonly the ones that need brain surgery. Taking L-dopa is used as the first line of defense against PD. In most cases just taking the medication is sufficient.

However, there is the risk that the effects of L-dopa wears off after taking the medication for an extended period of time. When this occurs the only remaining solution is to perform a Pallidotomy or DBS neurosurgery. A Pallidotomy is permanent, while a DBS is not. The procedural tasks are identical in both these operations. The aim is to determine the geometrical anatomical constraint of the GPi and then to disable the cells in the GPi. A Pallidotomy creates a lesion and a lead is implanted to regulate the GPi during a DBS operation. The MRI is used to calculate the path trajectory of the probe. The layers that the neurologist must identify are Striatum, GPe, GPi, Optic Track, Internal Capsule and Laminas. This identification is done by recalling the path trajectory, observing the cells that the probe encounters and through motor-sensory exploration. The ultimate goal is to accurately determine the location of the probe in the three dimensional space of the brain, and from that infer the geometrical constraints of the GPi. Currently most of these calculations, analysis and decisions are done in the mind of the neurologist. The Onetrack system provides new tools to the neurologist to facilitate this surgical procedure.

CHAPTER 3

DESCRIPTION OF "ONETRACK"

3.1 Introduction

The Onetrack system development is based on the microelectrode-guided Pallidotomy used at Emory University Hospital. The Onetrack system has been created to improve the overall accuracy and efficiency of the procedure. The system improves all areas of the operation from the brain imaging, probe trajectory calculations to the signal processing. New tools that are introduced through the Onetrack system include improved Magnetic Resonance Images (MRI's), a three dimensional model of the brain (3D-model), digital signal processing (DSP) and a Knowledge Based System (KBS). These Onetrack system components are discussed in this chapter, as well as the motivation for creating the Onetrack system.

3.2 Motivation for Onetrack

With new technological applications, comes new ways of improving current surgical methods. The Onetrack computer system consists of several technologically advanced components to improve a Pallidotomy or DBS surgery. Currently these procedures are performed using micro-electrode guided surgery [14]. This procedure involves the use of neurophysiological equipment to access and monitor neural activity in stereotactically targeted brain structures. The purpose is to first generate an exact neuro-anatomical map of the brain region and subsequently place thermo-coagulation lesions or a Deep Brain Stimulator (DBS) implant in a strategic location within this region. The Stealth^{TM} (The Virginia Spine Institute) [15] and Axon^{TM} systems (FHC) [16] are tools being used during this micro-electrode guided surgery. The Stealth^{TM} system is used during image-

guided surgery to interpret the MRI images [15]. The Axon^{TM} system is a tool for recording and viewing the neuronal signals [16]. The Onetrack system improves the operation by combining these systems and providing additional tools to the neurosurgeon and/or neuroscientist (from here on referred to as the user).

The goal of the Onetrack system is to improve the overall accuracy of probe location and efficiency of the procedure, thereby decreasing the amount of time the patient spends in the operating room. Currently the surgery can take 7 to 12 hours to complete and could be improved by automating some tasks. This could lead to the surgery being available to a greater number of people, and thereby reduce the present waiting time to have the procedure which, at some hospitals, is as much as several months. Another important purpose for the Onetrack system is to improve the safety of the operation. The tools in the Onetrack system give the user an improved visual description and analytical reasoning mechanism for the procedure, thereby, improving track planning and execution. The Onetrack system brings together all the tools used in the operation in one cohesive computer software program, so that the user has all the relevant information available during the operation.

3.3 Overview of Onetrack

Each component centers on a specific aspect of the surgery. Figure 3.1 shows the overall component configuration and interactions.

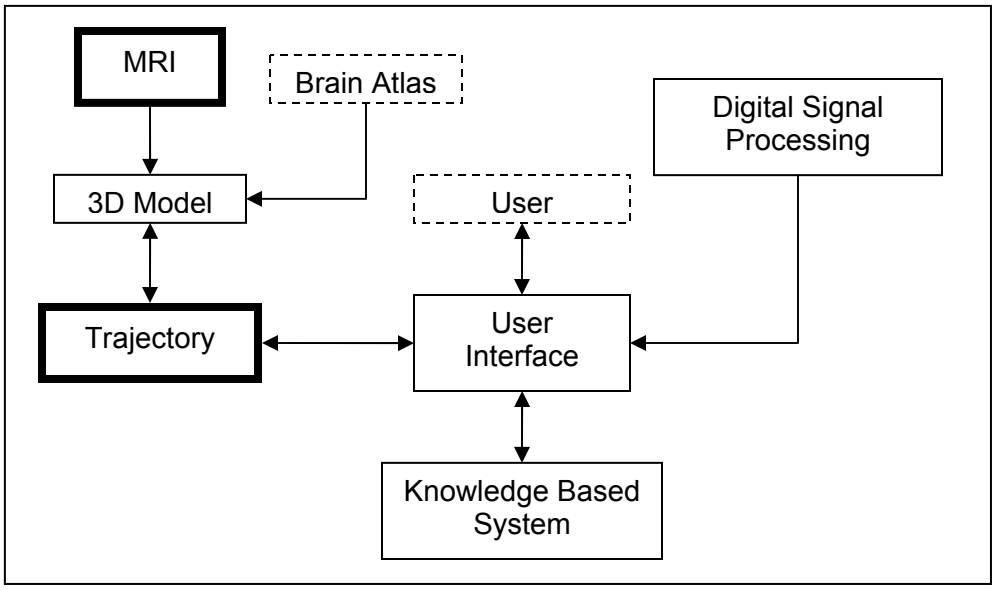


Figure 3.1. Onetrack component layout.

The surgical team (user) is in control, and therefore the User Interface is the command center for the Onetrack system. As the surgery progresses the user interacts with each of the components through the interface. For example, information from the 3D model to the KBS, must first pass through the user interface. The user box is indicated by a dashed line, indicating that it has not changed in the development of the Onetrack system. The light solid line indicates that the component of the Onetrack system is new to the procedure. The heavy solid line indicates enhanced components of the Onetrack system.

MRI images are taken prior to the operation. The information of the patient specific MRI and a generic Brain Atlas is processed by the 3D model and, with input from the user, a trajectory for the first track is determined. Once the track starts, the Digital Signal Processor (DSP) continuously analyzes the neuronal signal, and the Knowledge Based System (KBS) analyzes all the findings and determines the location of the probe.

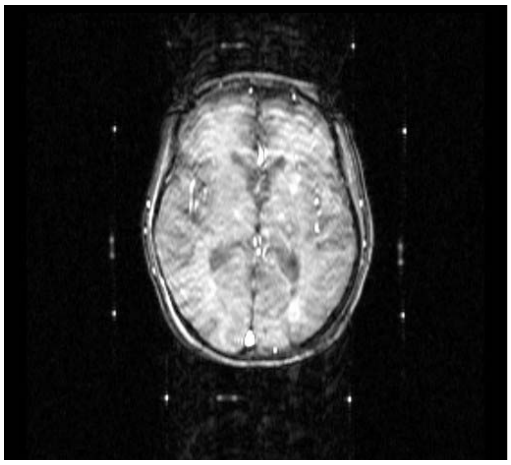
3.4 Onetrack Components

Each component has a unique task to accomplish within the operation, and correlates closely to the method currently being used. A description of each component follows.

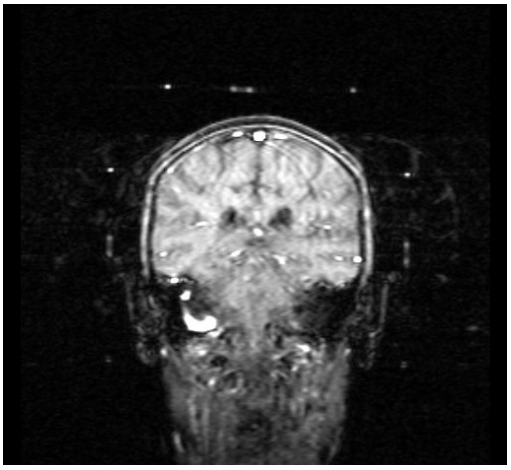
3.4.1. Magnetic Resonance Images

MRI's are taken of the patient's brain, with the stereotactic head frame in place, prior to the operation. The image slices are taken at approximately 3 mm intervals. Figure 3.2. shows a sample of such a slice. These images are compared to an anatomical model, such as the Schaltenbrand and Bailey atlas [12]. Using the Stealth^{TM} system [15], the user identifies the anatomical layers, using the mid commissural and mid-sagittal planes (shown in Figure 2.3.) as guide lines. After the target (GPi) is identified, the trajectory for the probe is calculated. The location of major arteries and other delicate structures are taken into consideration during this track planning phase. Part of the research being conducted for the Onetrack system, is the improvement of the resolution for these images, and also the refinement of the spacing between image slices. Currently details of the brain anatomy are being left out of the reasoning process with the three millimeters spacing and the Onetrack system is trying to improve these details. This has an affect on the path trajectory because the user has to interpolate between image

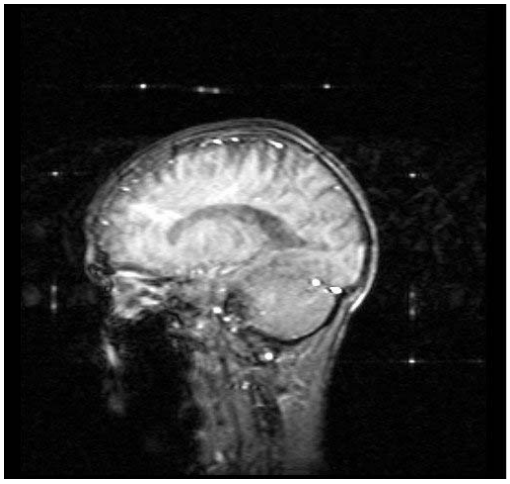
slices. The purpose of taking accurate MRI is to eventually morph the patient specific MRI with the brain atlas and generate a patient specific 3D model.



(a) Axial MRI view.



(b) Coronal MRI view.



(c) Sagittal MRI view.

Figure 3.2. Sample of MRI slices.

3.4.2. 3D Model Mapping

The 3D model uses the MRI images and brain atlas to generate the model. Figure 3.3. is a typical computer screen shot of the 3D model. The outer structure is the Striatum, the middle structure is GPe and the interior round structure is GPi as seen in the lower left quadrant of Figure 3.3. The user can select whether to view the 3D model represented in an isometric view (Fig. 3.3. lower right) or 2D cuts (rest of Fig. 3.3) parallel to the anterior-posterior, medial-lateral or dorsal-ventral planes. This gives the user flexibility in observing the location of the track in the virtual brain space.

One function of the 3D model is the generation of the path plan. As a track progresses, a depth reading (z-depth) of the probe tip is taken as it advances along the trajectory. This trajectory is an input into the Onetrack system. With this information the 3D model calculates the start and end z-depths of the primary structures (Striatum, GPe and GPi), the so-called predicted path plan. The predicted path plan is used by the KBS reasoning component during track execution. Another task accomplished with the 3D model is plotting the progression of the track along the path trajectory. This allows the user to visually gauge how the actual track differs from the predicted track. The 3D model also uses a matching algorithm to automatically find the best matches of the actual track with respect to the geometrical constraints of the brain [17]. This is done when the track is complete and the user selects the “automatic matching” feature. The user can then either select one of the matches or create his/her own match by clicking on the track and dragging it into position on the 3D model. The user is able to move around the actual track in the 3D model space. When the user is satisfied with the location of the actual track, a decision is made on where to place the next track. A new path trajectory is entered and the 3D model mapping process starts over by calculating the new path plan.

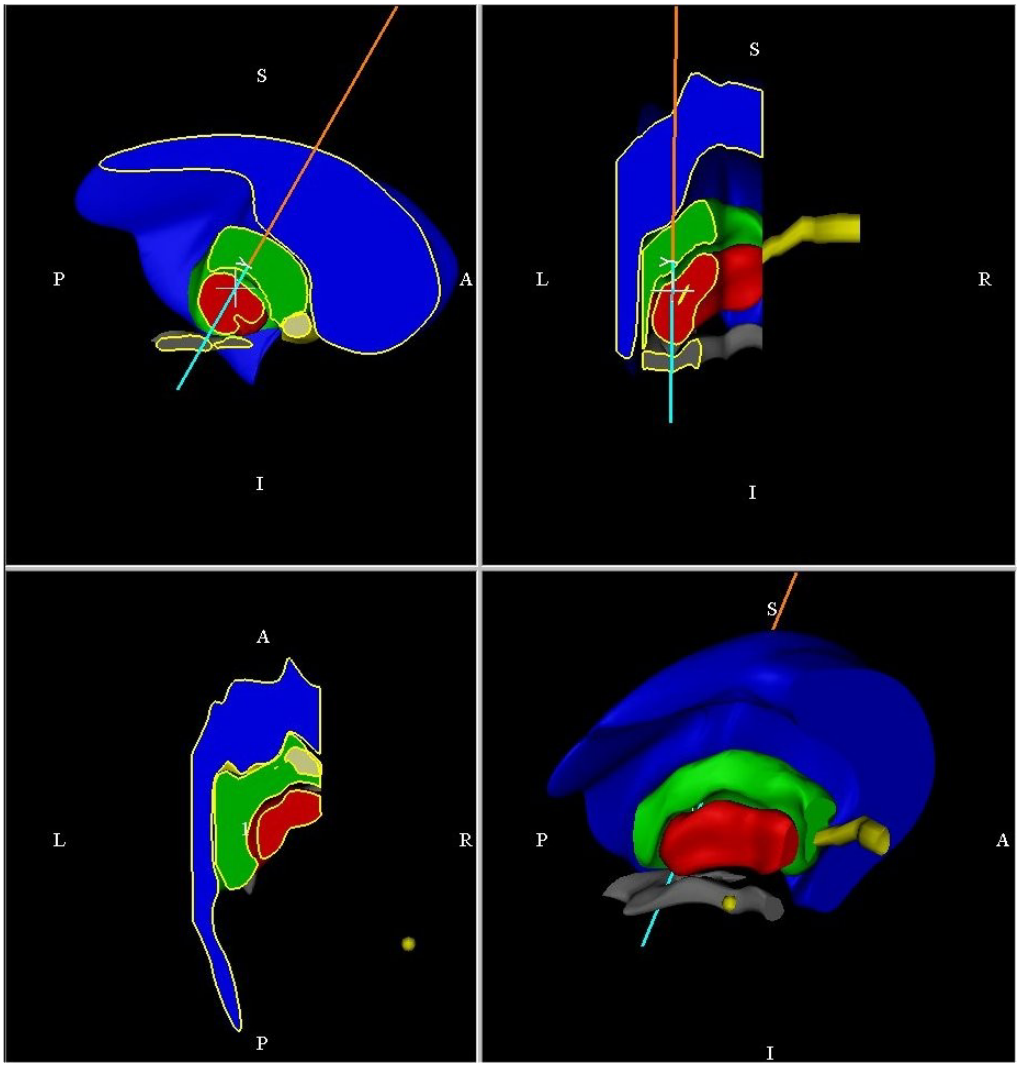


Figure 3.3. Computer screen shot of 3D model.

3.4.3. Digital Signal Processing

Currently the signal being produced by the probe is captured by the Axon™ system [16] shown in Figure 3.4. The Axon™ system allows the user to view the neuronal signal discharge on an oscilloscope type display. The Axon™ system also produces an audio version of the signal. The user identifies the neuronal cell by listening to and watching the neuronal discharge. Based on the frequency, amplitude, signal to noise ratio and pattern recognition of the signal, a trained user can distinguish between cell types.

The DSP component of the Onetrack system uses the Axon™ system to capture the signal and links it to the Onetrack system. The DSP component uses an algorithm to analyze the signal and determine a probability value for each cell type. Currently the DSP is only capable of determining the probability array for primary layers (Striatum, GPe and GPi). With refinement of the algorithm the DSP will eventually create a probability array for the cell type classifications.

Figure 3.4. Photograph of Axon™ system.

3.4.4. Knowledge Based System (KBS)

The MRI, 3D model and DSP components are based on pure algorithmic reasoning. However, with these components alone the Onetrack system can not accurately identify at which layer the probe tip is located. The reason for this is that the user's experience in using and interpreting the equipment is missing from the Onetrack system; the heuristic reasoning can not be captured in algorithms. Also, the KBS can reason with missing data. This is where the KBS is very effective.

The task of the KBS is to capture the heuristic reasoning of the expert and apply it to the Onetrack system. For example, the Striatum layer can never come after the GPe layer. The rest of this thesis focuses on the development of the KBS.

3.5 Summary

The tools in the Onetrack system provide the user with an improved visual description of the brain and analytical reasoning during the operation. The addition of these tools will reduce uncertainties, operating time, and the chance for infection while simultaneously improving overall efficiency and lesion or DBS placement accuracy. The components of the Onetrack system include the MRI, the 3D-model, the DSP, the KBS and user interface. The MRI provides patient specific images of the brain. The 3D model morphs the MRI images with the brain atlas and with the user inputs to determine the probe trajectory. The 3D model also generates the path plan, which is the start and end depth of each primary layer (Striatum, GPe, GPi and Optic Track) that the probe will encounter as it progresses along the trajectory. During track execution, the DSP is continuously analyzing the neuronal signal and generates a probability array of possible cell type selections. The user has the final say on the selection of cell type, and layer type for the

probe at a given depth at a specific time. The KBS guides the user and analyzes all information to assign the final cell type and layer type, at a specific depth at a certain time, to the Onetrack system. The next Chapter describes the different solutions that were investigated and the reasoning as to why the KBS is chosen.

CHAPTER 4

ALTERNATIVE SOLUTIONS

4.1 Introduction

Several probable solutions exist for the representation of knowledge involved in the Onetrack system. This chapter examines some of these solutions including mathematical models, neural networks, fuzzy logic and case based reasoning. A discussion is presented that include the benefits, drawbacks and possible implementations of each.

4.2 Mathematical Models

The first solution considered is the mathematical manipulation of all data involved in the Onetrack system. The main computer program of the Onetrack system is written in a mathematical modeling environment, so it is natural to investigate representing the expert's knowledge in the same environment. In a mathematical modeling approach and algorithm is developed to extract results from the data. "Normally, to develop a program to solve a particular problem, we first express the solution to the problem in terms of an algorithm and then develop a program that implements that algorithm" [18.] Such a program could present the user with results from which the user would determine the best course of action. In that case the Onetrack system would merely be an analysis tool that would present the user with feasible results that are based on an algorithmic manipulation of the data.

The benefits of a mathematical model are that the results are based on known mathematical truths. Thus the results produced by such a system would be accurate

and of high validity. Unfortunately the problem exists that results of such a model may have little meaning to the user.

One problem with using a mathematical model to represent the knowledge is the representation of linguistic constraints. For example, how would one represent that a popcorn cell is found in the Striatum layer only? Or how would one indicate that the GPe comes after the Striatum and the Lamina Exterior (Le)? The mathematical computations what would represent these statements are rather complex, because of the difficulty in representing the uniqueness of each patients conditions. The variability from patient to patient does not fit will into a mathematical paradigm. An easier solution must be sought to represent the decisions that the user makes during the operation. There is some intrinsic knowledge that the user has that can not be represented by an algorithm. Therefore it is necessary to consider alternate solutions.

4.3 Neural Networks

Another possible solution could be Neural Networks. Neural Networks fall under the category of Artificial Intelligence (AI). "Artificial Intelligence is the study of mental faculties through the use of computational models" [19.] A Neural Network aims to simulate the parallel reasoning capabilities of the human brain and is therefore one possible way to represent the knowledge involved in the Onetrack system. A generic definition of knowledge is given by Fischler and Firschein, 1987. "Knowledge refers to stored information or models used by a person or machine to interpret, predict, and appropriately respond to the outside world" [20.] A Neural Network aims to represent knowledge in a logical way, using computational models. There are two phases involved with creating a Neural Network. The first phase is learning, where "a subset of examples

is used to train the network by means of a suitable algorithm" [20.] Generalization or testing is the second phase when the trained network is presented with examples it has never seen before. The performance is assessed and if necessary the Neural Network is either retrained or its structure is modified. Different reasoning methods can be used in a Neural Network such as Mean-Least-Square algorithm and self-organizing systems. A detailed description of these are found in "Neural Networks: A comprehensive foundation" by Simon Haykin [20.]

There are several benefits to using Neural Networks. However, only two of these are applicable to the problem of representing decisions made by the Neurologist during a Pallidotomy or DBS. During pattern classification of the digital signal, produced by the probe, a Neural Network could identify the pattern and give a degree of confidence in the answer. The second benefit involves the fact that a Neural Network is based on a neurobiological analogy. The theory used to create Neural Networks is the representation of the brains activities. Therefore a Neural Network representation would be ideal to model the Basal Ganglia.

The Neural Network would be good for analyzing the probe signal. However there is another level of reasoning that takes into consideration several other input sources, such as the three dimensional models of the brain. One problem is that not enough examples exist to train the Neural Network and the answers are biased and based on the experience of the expert as opposed to an algorithm. The expert will always be present in the operating room, so the Neural Network could not make decisions in real-time independent from the expert. This defeats the purpose of having the Neural Network as the higher order reasoning mechanism. Also, the Neural Network is based on

algorithms, while the reasoning of the experts during a Pallidotomy or DBS is more of a heuristic and intuitive nature. Thus it becomes obvious that another solution needs to be found.

4.4 Fuzzy Logic

Fuzzy logic can be used in the modeling of linguistic uncertainty. It deals with the reasoning of approximate data as oppose to absolute known data. Therefore, it is a step beyond normal mathematical reasoning tools such as probabilistic methods. "Fuzzy logic, as its name suggests, is the logic underlying modes of reasoning which are approximate rather than exact. The importance of fuzzy logic derives from the fact that most modes of human reasoning and especially common sense reasoning are approximate in nature" [21.] One of the difficulties with the development of the Onetrack system is the representation of uncertainty. Uncertainty is found in the instruments that are used and the interpretation of the data measured by those instruments. "In general, uncertainty may arise from imprecise, inaccurate, or incomplete information, from verbal ambiguity, and from disagreement between different sources of information" [22.] Pure mathematical reasoning will allow for a margin of error in all of the data produced by these instruments. However, a Fuzzy Logic set could reason with and interpret these uncertainties. "Fuzzy logic is a universal technology; its applications range over the entire spectrum of electronic control" [23.] Therefore, it could be considered as one of the probable solutions to representing the knowledge in the Onetrack system.

Fuzzy logic has multiple benefits. Most importantly is the representation of knowledge in a mathematical environment. For example 50 °F might feel cold to someone from Africa but warm to someone from Antarctica. So the value of 50 can be part of two sets of data

depending on the point of view. "A fuzzy set suggests an item can have partial membership in two seemingly contradictory sets" [24.] Following is an excerpt from Zadeh in *Knowledge Representation in Fuzzy Logic* on the characteristics of a fuzzy logic.

> "Some of the essential characteristics of fuzzy logic relate to the following. In fuzzy logic, exact reasoning is viewed as a limiting case of approximate reasoning. In fuzzy logic, everything is a matter of degree. Any logical system can be fuzzified. In fuzzy logic, knowledge is interpreted as a collection of elastic or, equivalently, fuzzy constraint on a collection of variables. Inference is viewed as a process of propagation of elastic constraints" [21.]

In the Onetrack system fuzzy logic might be used to represent several activities or tasks that the user must perform in the course of an operation, such as analyzing the neuronal discharge signal. For example, a cell could be identified as either a pauser or a burster. The fuzzy logic set could identify which one of the two possible answers is more accurate than the other. So instead of saying that the cell is absolutely a pauser, the fuzzy logic could say that the cell is probably a pauser and only slightly a burster.

The fuzzy logic representation of knowledge is possibly the closest solution to the design problem. However there are some areas of concern. There is not enough existing data and knowledge to create a feasible fuzzy logic network. Too many unknown uncertainties exist within the Onetrack system. These first have to be identified and quantified before a reliable fuzzy logic network can be created. Therefore, an alternate solution needs to be found.

4.5 Case Based Reasoning

The general basis on which Case Based Reasoning (CBR) works is in the reasoning with previous known cases, comparing these cases to the current situation and then inferring a decision. “Case-Based Reasoning means to retrieve former, already solved problems similar to the current one and to attempt to modify their solutions to fit the current problem. The underlying idea is the assumption that similar problems have similar solutions” [25.] CBR is generally useful in modeling a general practitioners (GP) everyday reasoning. The GP may see a symptom in one patient that reminds him of what was seen in another patient, and therefore decides to use the same kind of treatment. “Case-based reasoning suggests a model of reasoning that incorporates problem solving, understanding, and learning and integrates all with memory processes” [26.]

The primary benefit of CBR is that the reasoning is based on previous proven solutions. Some interpretation and adaptation to match the current situation is needed, but the basis of the reasoning is sound. “CBR becomes more efficient by remembering old solutions and adapting them rather than having to derive answers from scratch each time“ [26.] The quality of the CBR depends on the following list (excerpt taken from the textbook *Case-Based Reasoning* by Janet Kolodner) [26]:

1. The experiences it has had
2. Its ability to understand new situations in terms of those old experiences
3. Its adeptness at adaptation
4. Its adeptness at evaluation and repair
5. Its ability to integrate new experiences into its memory appropriately.

The more cases the CBR is based on, the more adaptable the reasoning. In this project, the CBR could be potentially useful to reason with the previous procedures done on patients that suffer from Parkinson's. The CBR could reason with information available from previous track executions of the current patient.

Some of the drawbacks are that the CBR can not do much mathematical manipulation and finds it difficult to interpret information from other data sources (such as the other Onetrack system components). The reason for not using a CBR in the Onetrack system is that the knowledge of the expert does not resemble the CBR reasoning. The expert's knowledge is based on the current patient situation. Each patient's brain is unique in size, therefore it would be difficult to rely on reasoning with another patient's anatomical parameters. Also the procedure for each patient is unique and a lot of mathematical computation goes into the Onetrack system. The reasoning model needs to be able to interpret these data sources. For these reasons it is decided not to use a CBR in the Onetrack system.

Nevertheless, CBR might be useful to reason with previous tracks within the procedure of one patient. However, this is only one small portion of the bigger reasoning capability that the Onetrack system needs to have. The Knowledge Based System, described in the next chapter, is selected in the place of the CBR. The KBS resembles the reasoning capabilities of the CBR. However the KBS deals with more general knowledge than specific knowledge and is therefore more suitable for the Onetrack system.

4.6 Summary

Various probable solutions have their benefits and drawbacks. The mathematical model is good with mathematical manipulations but can not clearly represent the intrinsic rules in the knowledge of the expert. The neural network is good at combining intrinsic rules with the mathematical manipulations, but there are not enough cases and existing data sets to train a neural network. The fuzzy logic system should be excellent to represent the uncertainties within the system, however, it can not represent all the reasoning that goes into ultimately deciding what the layer and cell type of the probe is at a specific depth. Lastly CBR is based on reasoning with past known cases. The main problem with CBR is that the different patient cases do not relate to each other well, because each human's brain is unique in size and form. It is therefore decided to use a Knowledge Based System (KBS) with the combination of other components in the Onetrack system, to represent the knowledge of the expert.

CHAPTER 5

KNOWLEDGE BASED SYSTEMS (KBS) METHODOLOGY

5.1 Introduction

The design of a KBS is an iterative process that is completed when all the design requirements are met. The four phases of KBS design are planning, knowledge acquisition, coding and evaluation. Knowledge is gathered from an expert, encoded into a KBS by the knowledge engineer, evaluated and then redesigned as required. The interaction and importance of each phase is explained in this chapter.

5.2 Description of a KBS

The field of Artificial Intelligence is made up of subgroups, for example Neural Networks, Fuzzy Logic and Knowledge Based Systems (KBS). A Neural Network is a high powered number crunching reasoning tool that has the capability of training itself when information is missing or a new situation arises [23, 20]. Of the three, Neural Nets are the most adaptable mechanism during real time execution. Fuzzy Logic is used where there is high uncertainty and some mathematical manipulation [23]. KBS has very limited mathematical manipulations, is non-algorithmic and is primarily based on a set of heuristic reasoning rules [27].

These rules are obtained from either an expert in the field of interest or books containing relevant information. An expert is a person that has working knowledge on how the task is performed. For the Onetrack system a neuroscientist (Expert 1) and a neurologist (Expert 2) were approached to provide the necessary knowledge. These rules are encoded into the KBS and become part of the knowledge base.

The interaction between the user and the KBS works as follows; the user presents the facts to the KBS and the KBS provides expertise back to the user. Figure 5.1 show this interaction. The KBS consists of three primary parts, the knowledge base, the context and the inference engine.

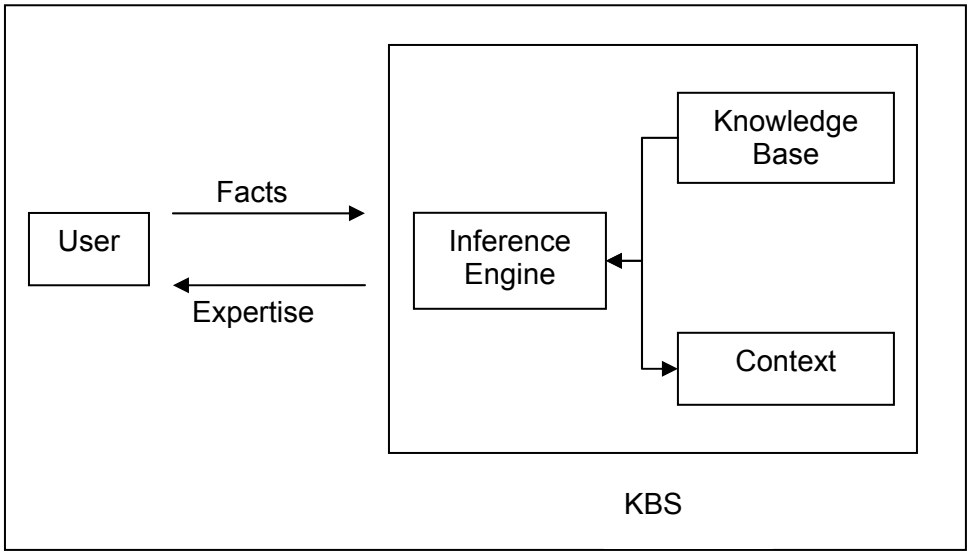


Figure 5.1. User and KBS interaction.

5.2.1 Knowledge Base

The Knowledge Base consists of the

(a) Domain Knowledge.

This knowledge does not change because it is based on absolute facts. For example, the order in which the probe encounters the different anatomical structures along a track.

(b) Process Knowledge.

This knowledge controls the processing of rules in the KBS. For example, the KBS needs to be initialized before a track begins. The process knowledge adapts as more knowledge is gathered and the program develops.

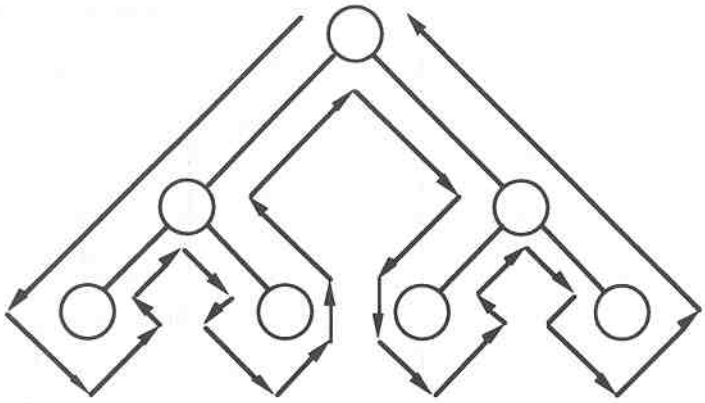
In the development of the Onetrack system the process knowledge changes over time, as more knowledge about the operation is gathered. Sometimes the entire KBS program is reconstructed if the process knowledge changes. However, the domain knowledge remains constant through all the modifications, which simplifies any reconstruction of the KBS. The context part of the KBS is generic for the problem context in which the KBS is being implemented.

5.2.2 Inference Engine

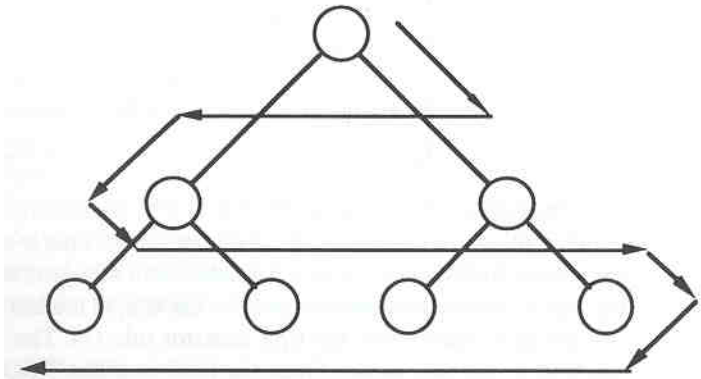
The inference engine determines the steps that are taken (rules that are activated) to determine a conclusion for a given problem. "The knowledge base contains the knowledge with which the inference engine draws conclusions. These conclusions are the expert system's *(KBS's)* responses to the user's queries for expertise" [27, Chapter 1]. When the KBS is running, it poses questions to the user from time to time, and the user's response affects the conclusions drawn by the KBS. The inference engine takes the user's response (a fact) and matches it to the knowledge base rules. The selected rule governs the outcome of the conclusion.

Two typical types of inferences used are depth-first reasoning and breadth-first reasoning. "In a depth-first search, the search goes down as far as possible and then back up. A breadth-first search proceeds one level at a time before descending to the

next lower level" [27, Chapter 2]. Figure 5.2 illustrates the depth-first and breadth-first reasoning for an arbitrary decision tree [27, Chapter 2]. The arrows represent the order to visit the nodes in and the circle represent nodes of specific knowledge. The Onetrack system is a depth-first reasoning mechanism because the most recent information and rules are taken into consideration first. The KBS must be capable of coming to a conclusion quickly and be able to function adequately in the absence of information.



(a) Depth-first Search



(b) Breadth-first Search

Figure 5.2. Depth-first and Breadth-first Searches for an Arbitrary Tree [27].

The following are two examples of how a KBS would infer a conclusion based on the fact from the user.

Example 1

This is an example is based on a similar example given in *Expert* Systems written by Giarratano [27] and presents the fact and rule statements that can be found in a KBS.

Fact 1: The banana hangs from the ceiling.

Fact 2: The ladder is against the wall.

Rule 1: If the monkey is in the same room as the ladder, then the monkey moves the ladder to the bananas.

Rule 2: If the ladder is under the bananas, then the monkey gets the banana.

Rule 3: If the monkey has the banana, then the monkey eats the banana.

The user would tell the KBS Fact 3: The monkey enters the room.

KBS would infer Fact 4: The monkey eats banana.

Example 2

Rule 1: If it is raining then take umbrella.

Rule 2: If sun is shining then do not take umbrella.

User would tell the KBS Fact 1: Sun is shining.

KBS would infer Fact 2: Do no take umbrella.

Note that no mathematical computation is involved with either of these examples. Also, it would be very difficult or nearly impossible for large systems to effectively encode these examples in an algorithm. This makes the KBS powerful in heuristic reasoning, and applicable to use in the Onetrack system.

Other benefits for using a KBS are

a) Multiple expertise.

   Multiple sources containing expert knowledge can be made to function in a concise manner. The combined level of expertise can exceed that of an individual expert.

b) Performance.

   The KBS is permanent and as long as the domain knowledge and process knowledge remains constant, the KBS is constant. The KBS can not forget, or retire like human experts.

c) Reliability.

   The KBS can increase the reliability of a decision made by the expert. It can even correct an expert, if the expert neglected to take some information into consideration during reasoning.

d) Construction.

   The ease of constructing a KBS makes it easy to break down the programming tasks. It is not necessary to encode the KBS in sequential order from start to finish. The KBS is flexible and the middle section can for example be coded before the front section.

e) Reasoning.

   The KBS is capable of reasoning in the absence of information and in the presence of uncertainty.

The knowledge base and inference engine work together to provide the user with accurate conclusions, based on expert knowledge.

5.3 Design process of a KBS

The KBS in the Onetrack system is based on the incremental waterfall model [27, Chapter 6]. The KBS is developed by incrementally adding one function after the other. “The primary advantage of the incremental model is that the increases in functional capability are easier to test, verify and validate” [27, Chapter 6]. This then also allows for incremental testing of the KBS.

The design phases for the development include planning, knowledge acquisition, coding and evaluation, illustrated in Fig 5.3. This is an iterative process. Each phase is described below.

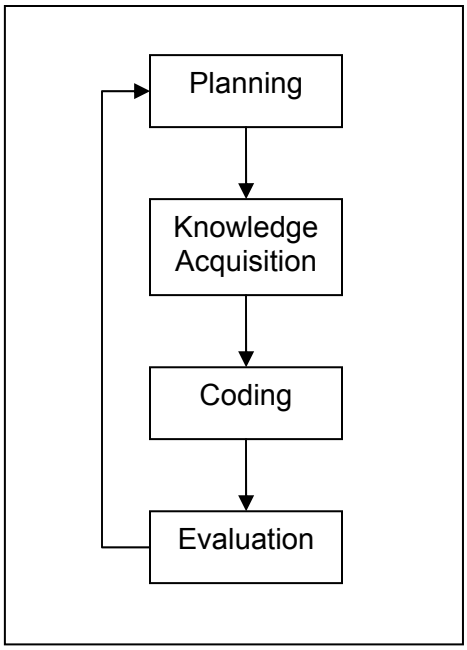


Figure 5.3. Life cycle of KBS.

5.3.1. Planning

During the planning phase the scope of the KBS is determined. Decisions are made about what the KBS is required to do and how the KBS interacts with the rest of the Onetrack system. A list of assumptions is made to narrow the scope and focus the requirements of the KBS.

5.3.2. Knowledge Acquisition

In Knowledge Acquisition (KA) rules are gathered for the knowledge base. This is accomplished by reading books and journals on the topic, interviewing experts and attending project update meetings. The development of the entire Onetrack system affects the implementation of the KBS.

5.3.3. Coding

Rules are gathered by closely examining the knowledge gathered. Generic facts such as cell type to layer combinations are encoded. Also the knowledge is examined to determine the process knowledge. Both the domain knowledge and process knowledge are encoded for a specific activity. Once an activity has passed initial testing, then the next activity is encoded. This is done iteratively until the KBS contains all the relative tasks. Part of coding involves the integration of the KBS with the Onetrack system. This is also done incrementally until all the system functionality is available.

5.3.4. Evaluation

Once the KBS is stable, it is evaluated to determine how well it meets the planning phase requriements. During the evaluation areas of weakness in the KBS are identified.

Sometimes this forces the knowledge engineer to completely re-engineer the process knowledge of the KBS, thus planning starts over.

This loop can go on indefinitely as more knowledge can always be added to the KBS. However, at some point the decision is made by the knowledge engineer that the KBS satisfies the design requirements, and the process is stopped.

5.4 Summary

The KBS is not an algorithm and thus has limited mathematical capability. Its strength lies in its reasoning capability, which can not be captured by an algorithm. The knowledge base is made up of the domain knowledge and process knowledge. The domain knowledge is stable and based on known truths, while the process knowledge is flexible. The process knowledge evolves until it closely resembles the actual procedure an expert uses to come to conclusions. The inference engine matches all facts to rules, is responsible for the activation of all rules, and is therefore the true reasoning mechanism used in the KBS. For the Onetrack system the KBS is a depth-first reasoning mechanism. Once the scope of the project is determined during planning, knowledge acquisition starts. The knowledge gathered is analyzed and knowledge statements are extracted. The code is tested, evaluated and refined. Once the program produces results close to the desired outcome set forth by the scope, then further development is stopped. If not, then planning starts again, and it is possible to refine or change the original scope. In this research the scope for the Onetrack system has already been defined and is described in chapter 7. The next Chapter discusses Knowledge Acquisition (KA).

CHAPTER 6

KNOWLEDGE ACQUISITION

6.1 Introduction

Knowledge Acquisition (KA) is often times a long and tedious process. From 2001 to 2003 knowledge was gathered for the KBS in the Onetrack system. A variety of methods were used, including Classical KA and Protocol Analysis. Through literature review and interviews the process knowledge and domain knowledge of the KBS have been established.

6.2 Purpose of Knowledge Acquisition

Knowledge is the foundation of any KBS. "The main objective of the knowledge acquisition task… is to produce and verify the knowledge required by the system" [28, Chapter 7]. Through interviews, and examining literature, the knowledge engineer can create the KBS. Every additional new piece of knowledge can potentially be a rule or fact in the KBS. The more rules and facts in the KBS, the more accurate the KBS conclusions will be. It is the knowledge engineer's responsibility to decide what level of detail is necessary to meet the design requirements.

KA is the formal method by which the knowledge engineer gathers knowledge. In stand alone KBS systems it is sufficient to only include knowledge from experts and journals written by experts. However, in integrated KBS systems, such as the Onetrack system, it is also necessary to consider the tasks, purposes and functions of the other components in the Onetrack system. All of the components are integrated to make up a single software application.

However, a dilemma arises as it is difficult to determine when there is sufficient knowledge in the KBS. The knowledge is gathered from the expert, implemented into the KBS and then integrated with the software application, as illustrated in Figure 6.1. The expert uses the application and new procedural knowledge is created, which starts the entire process over. There is always more knowledge that can be included in the KBS. Thus it is impossible to ever create a KBS that completely contains every single detail, down to the molecular level, of the decision process that it is imitating.

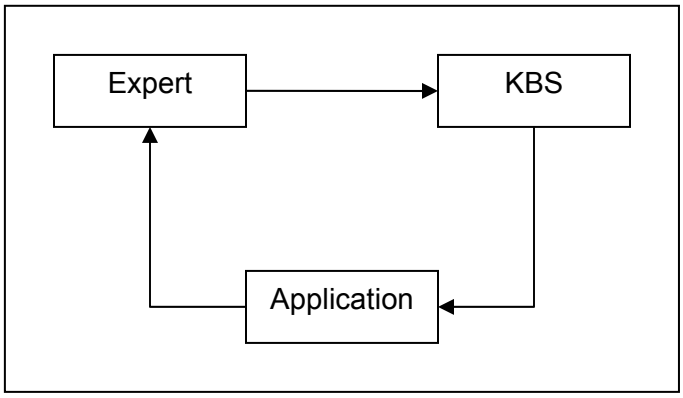


Figure 6.1. The Knowledge Acquisition loop.

The three approaches of KA used to obtain knowledge, are Classical KA, Protocol Analysis and Retrospective Evaluation.

6.3 Classical Knowledge Acquisition

The first step in acquiring knowledge is called Classical KA. Literature reviews are undertaken to gather basic ideas and building blocks in the general topic area. For the Onetrack system, this included literature on the nervous system [1], neurosurgery for PD [3], and on PD itself [5].

The knowledge engineer also interviewed Expert 1, in an informal setting. This was done to obtain information on how the procedure is performed and to gather some basic anatomical truths. Some of the knowledge gathered during this interview has not changed during the Onetrack KBS development. For example, the possible orders in which the probe will find the anatomical layers of the brain. The probe can never find the GPe first and then the Striatum. This fact is encoded in the KBS.

On one occasion, the knowledge engineer was able to attend a surgical procedure. This gave the knowledge engineer a first hand account of all the activities that take place during the operation. By understanding the relation between the activities and the decisions that stem from the activities, the knowledge engineer is able to comprehend how to encode the process knowledge of the KBS.

Another method undertaken for gathering knowledge was listening to recordings taken during actual operations. The recordings included the signal from the Axon^{TM} system [16] and the audio of the neurologist's comments. Data was collected by comparing the notes taken by a technician during the operation and comparing them with the audio comments of the neurologist. This provided the knowledge engineer with some test

cases for the KBS and aided in determining the inputs and outputs that the KBS could expect.

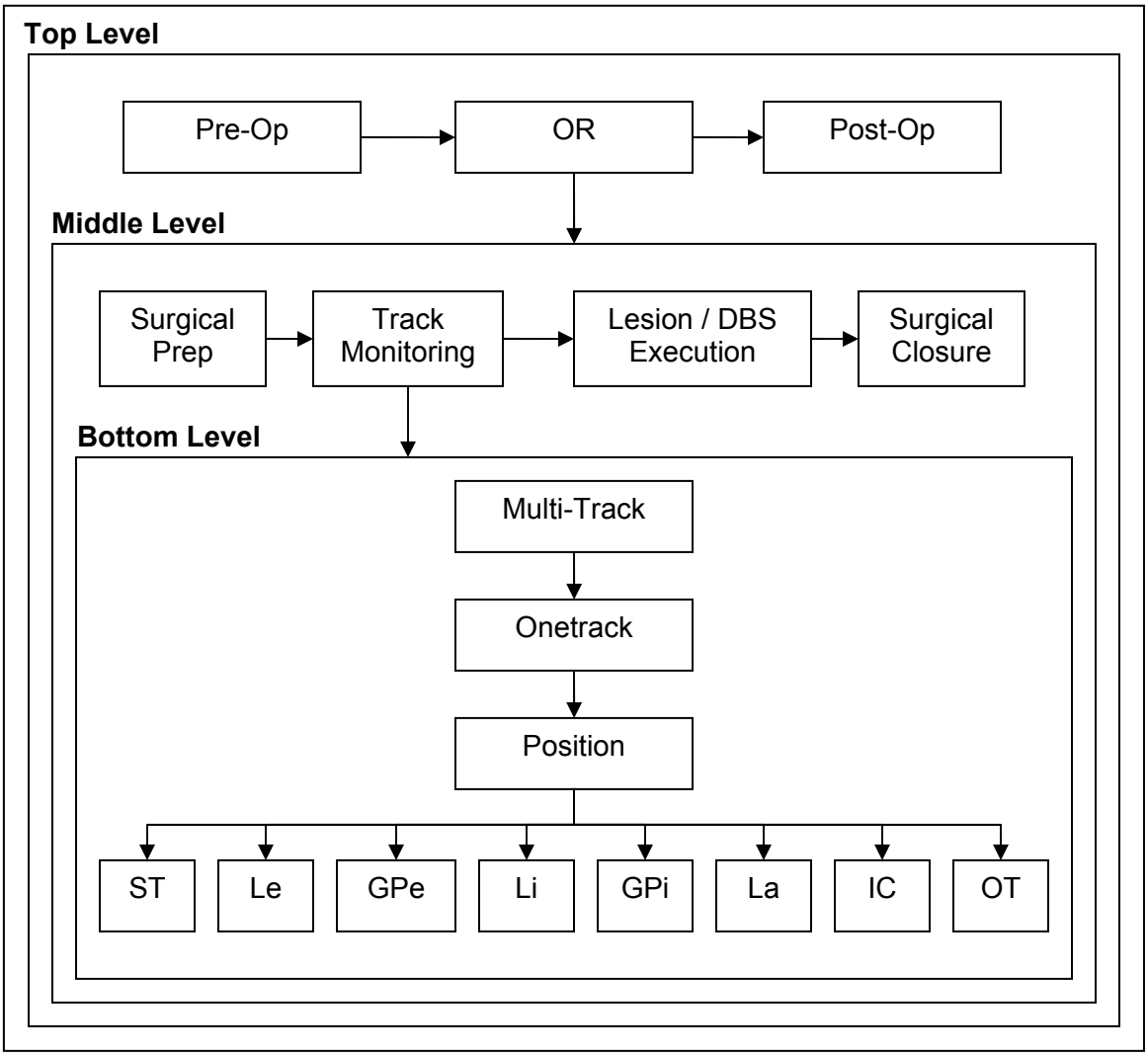


Figure 6.2. Original process knowledge.

Upon completion of the Classical KA, the knowledge was analyzed and encoded. As more knowledge became available, the KBS continued to be modified. The original KBS is a backward chaining process knowledge, as indicated in Figure 6.2. "Backward chaining involves reasoning in reverse from a hypothesis, a potential conclusion to be proved, to the facts which support the hypothesis" [27]. Reasoning would start in each anatomical layer (ST, Le … OT), by looking at anatomical facts, audio processing and patient interaction. By examining a given set of inputs the KBS would then determine the potential layer. These decisions then make up the Onetrack KBS reasoning procedure. Multiple-Tracks would consist of multiple Onetrack's that were successfully completed. These steps make up the Bottom Level of reasoning that is part of the Track Monitoring, which is a phase in the overall operation. Originally the scope of the project was to have a KBS that aided in guiding the user through the Middle level of reasoning which included surgical preparation, track monitoring, lesion or DBS execution and surgical closure. However it was decided to narrow the scope by focusing on track execution. The KBS would then ultimately also include the Top level of reasoning which considers pre operation tasks and post operation tasks. All of these functions could ultimately be added to the KBS system. However, it is necessary to focus the scope of this project to the monitoring of the actual track.

As the Onetrack system started to take shape, it became apparent that the KBS was lacking in some areas and that the process knowledge might be an inaccurate representation of the actual procedure. The KBS therefore needed to be more in sync with the other components in the Onetrack system. Problems arose as to how information should be passed to and from the KBS. Lastly, it was decided that the

original scope of the KBS was too broad. The KBS needed to be refined to only considering what occurs during the operation and more specifically during the execution of a track. More knowledge of the track execution needed to be gathered. Therefore, it was decided to add Protocol Analysis in order to validate current knowledge and gather new knowledge.

6.4 Protocol Analysis

Protocol Analysis is done by conducting a formal, structured and recorded interview with the expert. There are two goals for using verbal reports, transcripts of interviews, in the development of a KBS.

> "The first goal is to elicit and extract knowledge and rules from experts in order to build computer models of expert performance (expert systems) that can aid or even replace experts in everyday life for a wide range of tasks. The second goal is to study superior expert performance on specific tasks under controlled conditions and to assess the cognitive process, knowledge and acquired mechanisms that mediate the superior performance of experts" [18, Preface].

The KBS (expert system) in the Onetrack system is created to aid the expert, and not to replace the expert. The user will always be at the center of the process because no computer program can capture all knowledge needed. The doctor is always in control of the entire Onetrack system. The verbal reports of the experts help to assess the cognitive process knowledge.

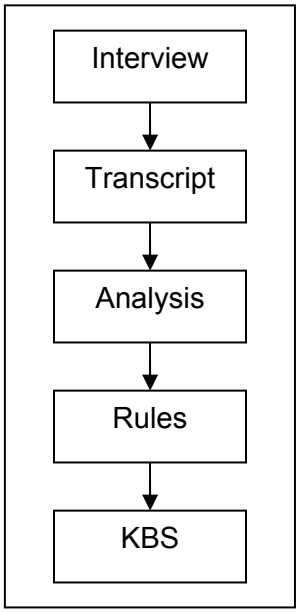


Figure 6.3. Process of rule generation in the KBS.

The process involved with obtaining the rules to encode in the KBS is illustrated in Figure 6.3. First the expert is interviewed. An analysis is done of the transcript, which is made from the taped interview. The analysis leads to the creation of rule statements. These rules are then encoded using the KBS terminology and thus the KBS is created.

6.4.1. Controlled Conditions.

The conditions of the interview have to be controlled in order to obtain objective and valid knowledge from the expert. The interview is video taped so that the audio and visual behavior of the expert are captured. The camera is placed in front of the expert to capture facial expressions and hand gestures.

The expert is presented with written instruction on how to complete the interview. A list of questions spaced out to allow room for writing is given to the expert. Once the interview starts the interviewer says little to nothing to the expert. The purpose of this is to obtain an unbiased response from the expert.

The expert talks aloud explaining his thought process, while completing the tasks. “When we are interested in preserving the temporal properties of the cognitive process we may ask our subjects to talk aloud” [28, Appendix]. The interviewer remains silent throughout the process except to remind the expert to continue talking aloud.

6.4.2. Procedure.

The expert is given written instructions that have to be read aloud. These instructions explain the purpose of the interview and the different tasks that need to be completed. The interview is divided into three tasks: warm up, Protocol KA and Retrospective KA. Appendix A contains the documents that were presented to the expert.

a) Task 1: Practice Session.

The practice session is used as a warm up, to get the expert used to talking aloud. The purpose of doing this is to get the expert comfortable with expressing his/her thoughts verbally. "The warm-up tasks are selected to be particularly easy to use "talk-aloud" with and one's for which the cognitive processes are well understood" [28, Appendix]. An anagram problem, found in Appendix A, was used as a warm up for the Onetrack KA.

Two experts are interviewed for the Onetrack system. Expert 1 had a difficult time solving the anagram, because English is not his first language. However, the exercise still serves its purpose because the expert is vocally expressing his thoughts.

b) Task 2: Walk through of a track.

Protocol KA is used in this task. The expert is presented with data from an actual case and asked to analyze the procedure. Specific written questions are asked to prompt the expert. The questions are open ended and leave much room for interpretation. The purpose is to capture knowledge concerning the overall procedure followed during the operation. Both the experts interviewed for Onetrack

provide valuable information concerning the procedure. This is reflected in the new process knowledge of the KBS. Also the knowledge obtained from both experts is similar in nature.

c) Task 3: Follow up questions.

This task is based on Retrospective KA. "A durable (if partial) memory trace is laid down of the information needed successively while completing a task" [28, Chapter1]. It is best to complete task 3, right after task 2. The reason for this is that the decisions and reasoning from task 2 are still fresh to the expert. Task 3 contains pointed and direct questions that refer to specific aspects of the operation. This serves to help clarify some aspects of the procedure. The questions are short and the answers should be short as well. Here the expert is not solving a given problem, but instead explaining a certain aspect of the procedure, so the reasoning is different. Some of the answers might be repetitive, but it is still valuable to determine whether the expert's answer remains the same.

Only Expert 1 completed task 3 for Onetrack. An interview with Expert 2 was not able to be scheduled so task 3 could not be completed. However, it was decided that there is sufficient similarity between the expert's answers in task 2, and enough information was available to move on.

Adequate conclusions can be drawn by combining the knowledge gathered by task 2 and task 3. Transcripts of the interviews can be found in Appendix B.

6.5 Analysis

The transcript of the interview is then analyzed to extract the rules. When analyzing the transcript, the knowledge engineer searches for key words, such as: so, then, therefore, because… These words signify that either a conclusion follows or reasoning about a conclusion follows. These words help break up and identify key sentences in the transcript. Appendix C is the combined rules extracted from both experts. These rules are then evaluated to determine their level of importance and relevancy to the KBS. After determining the final set of rules, these are encoded into the KBS.

When analyzing papers, journals and other such documents, it is best to look for explicit details that the KBS might be lacking. The literature review serves to verify and compliment some of the knowledge gathered during the interviews.

Lastly knowledge is gathered during project meetings and analyzed to determine how to incorporate it into the KBS. This knowledge directly influences the role and purpose of the KBS. It also affects what information is passed to and from the KBS. This knowledge is also analyzed to determine how to incorporate it into the KBS.

The current process knowledge (also the Intermediate KBS, Chapter 7) is reflective of the knowledge that was gathered during Protocol Analysis, indicated in Figure 6.4. This reflects a more refined role of the KBS in the Onetrack System. The process knowledge is no longer backward chaining, but instead a combination of two sequences. The inner loop deals with the execution of a single track, and the outer loop deals with the execution of multiple tracks. Also a Wakeup mechanism, that activates the KBS reasoning, is incorporated into the inner loop that notifies the KBS when new information

becomes available. The Sleep mode is when the KBS is completed processing and is waiting for further input from the user. This current process knowledge is more sensible and fits in with the rest of the Onetrack system components.

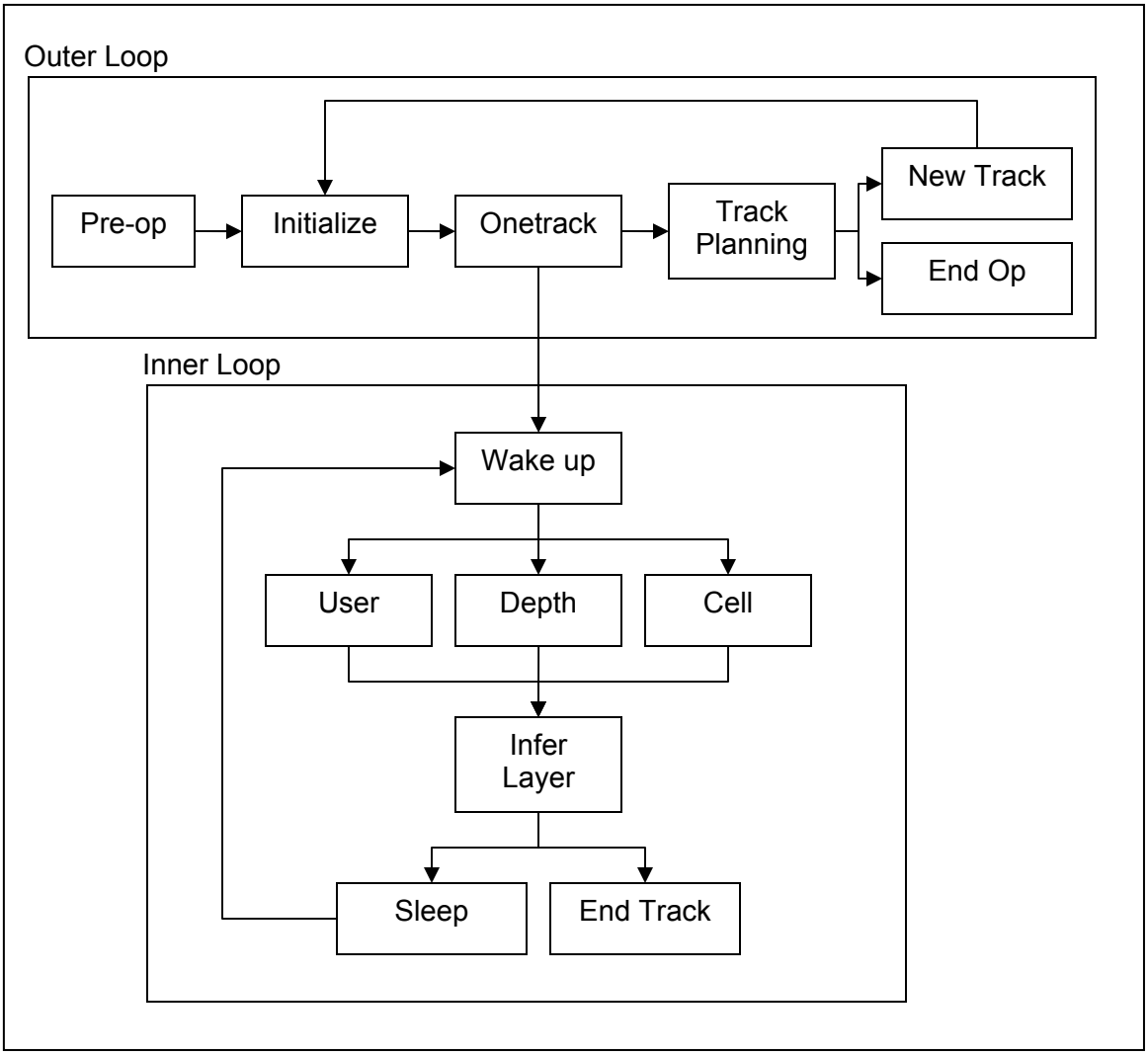


Figure 6.4. Current Process Knowledge.

6.6 Retrospective Evaluation

Once the Onetrack system is completely integrated and stable the expert will be allowed to interact with the system and practically evaluate it, like a “test drive”. The expert will be asked for an opinion on how it can be improved, and whether something is missing and/or incomplete. This will allow the knowledge engineer to evaluate the KBS and determine where the areas of weakness are and whether something can or should be done.

6.7 Summary

The knowledge engineer has to decide when sufficient knowledge is encoded into the KBS. The methods that have been used to gain knowledge include Classical KA and Protocol Analysis. Classical KA includes literature reviews, informal interviews, observing the operation and listening to recordings of the operation. This leads to the creation of the original process knowledge. As the Onetrack system components are starting to integrate it becomes apparent that the process knowledge in the KBS is an inaccurate representation of the procedure. Thus it was necessary to do Protocol Analysis to validate current knowledge and identify new knowledge. This takes the form of a formal and video taped interview which is transcribed and analyzed before it is encoded into the KBS. The next chapter deals explicitly with the knowledge that is encoded in the KBS.

CHAPTER 7

KBS CODE

7.1 Introduction

Knowledge Acquisition produces a data set of possible knowledge that can be encoded. The knowledge reflected in the KBS in this research is based on procedures performed at the Emory University Hospital in Atlanta Georgia. This knowledge can be categorized into two groups. The first group is process knowledge. Process knowledge contains all relevant knowledge pertaining to the order of reasoning activities of the KBS and the knowledge of the surgical team in how to decide what step is next to accomplish. The second group is domain knowledge. The functional description of each activity reflects the domain knowledge. Process knowledge and domain knowledge comprise the encoded knowledge. The process of creating a KBS is iterative. With every iteration a new capability is added to the reasoning and the interaction of the code becomes more complex. Due to this complexity it is necessary to define the scope of the KBS. For this project the KBS is meant to identify the layer and cell types for the probe at a given depth at a certain point in time and to suggest to the user where the next track should be placed once the current track execution is completed.

7.2 Scope

The research objective is to create a stable KBS that will serve as an analysis and guidance tool for the Onetrack system. The surgical procedure is complex, involving various decisions continuously being made by the user. At various moments in time the KBS aids the user by presenting a possible solution based on the input sources that can

be analyzed at that point in time; from one to all sources. Ultimately the user's input and response to the KBS takes precedence over all the decisions that are made by the KBS.

To understand the purpose of the KBS, it is important to understand the information flowing into and out of the KBS as illustrated in Figure 7.1. Discussion follows below.

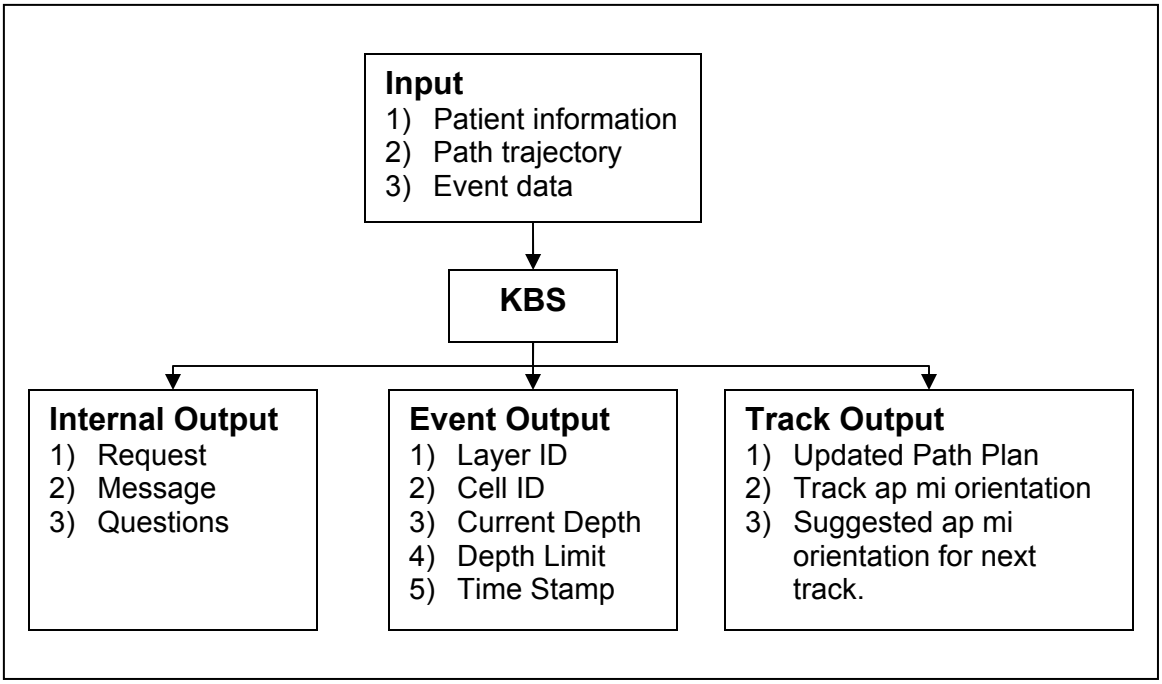


Figure 7.1. Schematic of the input and output information for the KBS.

7.2.1. Input to KBS

There are three situations that constitute information being sent to the KBS. This is when the Onetrack system provides the KBS with:

1) general patient and operation information
2) the predicted path trajectory, or
3) event information

An event is one of the following situations:

a) The user inputs new information about the track.

b) The DSP generates an array of probable cell types at a specific depth.

c) A depth limit, specified by the KBS, is reached.

All events occur during track execution. A track is considered to be the part of the operation when the probe is in the brain. The 3D model component of Onetrack system provides the predicted path trajectory to the KBS while the DSP component provides the array of probably cell types and the depth reading of the probe. General patient and operation information is entered by the user at the beginning of the procedure, and this information is passed to the KBS.

7.2.2. Output from KBS

The output from the KBS is divided into three groups, namely Internal Output, Event Output and Track Output.

1) Internal Output.

The KBS notifies the Onetrack system when more information is needed to complete the KBS reasoning tasks. In some instances the KBS requests specific information from other components in the Onetrack system. The KBS can post a message to the

user or question the user about a specific input variable. This ensures that the answer obtained from the user has high validity in the KBS.

2) Event Output.

The primary objective of the KBS is to classify the layer and cell type at a specific depth and point in time. Event classification is based on the knowledge of the KBS and depends on the event information input. The layer and cell type classification becomes available when the KBS completes reasoning with the event information. The KBS notifies the Onetrack system of the next depth at which it needs information. The depth limit ensures that the KBS is sent information at specific depths along the predicted path trajectory.

3) Track Output.

As the probe progresses along the path plan, the depths at which layers are encountered are recorded. At the end of a track the KBS generates a new path plan. The KBS is able to suggest where to place the next track, depending on the results from the new path plan.

The ultimate goal of the KBS is to notify the user where in the brain the probe is, for safety and mapping of structures. The Event Output and Track Output define the objectives of the KBS for any Pallidotomy or DBS procedure. The next section describes the sequence of activities involved to produce these outputs.

7.3 Implementation Strategy

Implementing the KBS involves looking at the results from knowledge acquisition and answering the following two questions:

1) How closely does the KBS reflect the decision making process of the expert?
2) How does the KBS relate to the functions of the other components in the Onetrack system?

A fine balance exists between the implementation of the answers to these two questions. It is necessary to optimize the information from the expert while optimizing the interaction between the KBS and the other Onetrack system components. Therefore the KBS has three levels of functionality; minimum, intermediate and ultimate.

7.3.1. Minimum KBS

For the KBS to be of any use, a minimum functionality must be achieved. The KBS must be able to reason using the depth of the probe, compare this value to the predicted path trajectory and identify the layer type. To achieve this, the process knowledge contains two sequence loops, an outer loop and an inner loop. The activities involved with each loop are shown in figure 7.2.

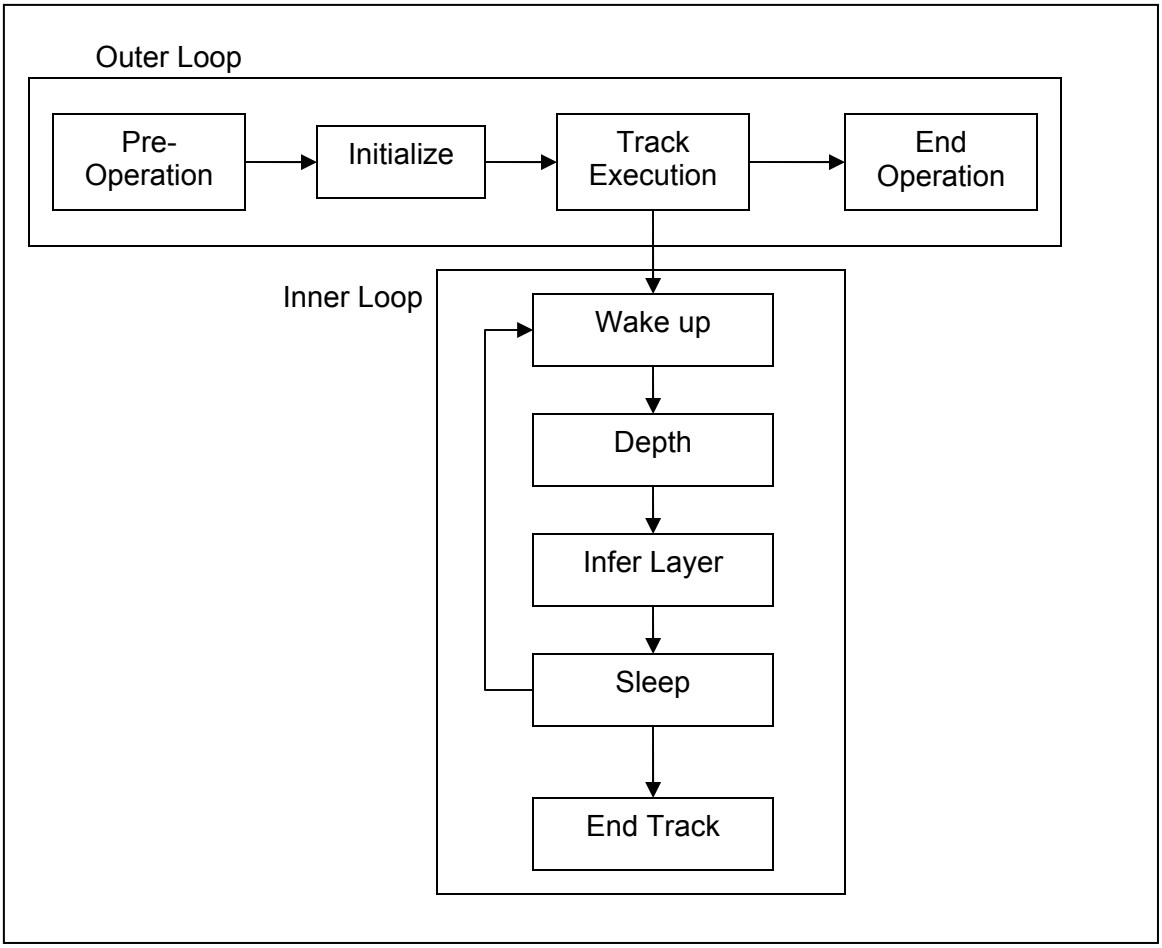


Figure 7.2. Minimum KBS

The outer loop contains activities that occur before and after the track execution. General information concerning the patient and the operation are sent to the KBS during the *Pre-Operation* activity. During the *Initialize* activity the KBS receives the predicted path plan from the 3D Model. The KBS is now ready for *Track Execution.* After the completion of the *Track Execution* activity, the user can either start another track or end the operation.

The inner loop contains activities that occur during *Track Execution.* Event information is sent to the KBS during the *Wake up* activity. In the minimum KBS, only the depth of the probe tip with a time stamp is an acceptable input. The timestamp uniquely identifies the information. During the *Depth* activity the current depth of the probe is compared to the path plan and a probable layer is identified. In the *Infer Layer* activity the final layer is determined and in the *Sleep* activity the answer is passed to the Onetrack system. The user can then either end track execution or create a new event.

In each of the sequence loops the current activity is dependant on the completion of the preceding activity. If for any reason the preceding activity was unsuccessful, the KBS will halt. The only way to ensure that this does not happen is by debugging the code. This concludes the discussion on the process knowledge for the minimum KBS.

7.3.2. Intermediate KBS

The Intermediate KBS is built (also current KBS, Chapter 5) on the foundations of the Minimum KBS. Figure 7.3 shows the process knowledge layout for the Intermediate KBS. The activities added to the KBS are *User* input, *DSP* reasoning and *Track Planning*.

The outer loop remains a sequence, with the addition of a new activity called *Track Planning*. As the track progresses the KBS notes the depth value at which each layer is actually found. After *Track Execution* the KBS produces the actual found path plan (new path trajectory). During *Track Planning* the actual path plan is analyzed, and based on a set of knowledge statements, the KBS then suggests where to place the next track. However, the user is not required to follow this guidance.

The inner loop is changed from a simple sequence to a combination of sequence and backward chaining reasoning method. The sequence is composed of activities that consist of *Wake up, Infer Layer,* and *Sleep.* The *DSP*, *User* and *Depth* activities form a backward chain that feeds into the *Infer Layer*. This feature adds flexibility in that not all three activities in the backward chain have to produce valuable information in order to identify the layer. Currently the KBS allows three combinations, either *Depth* activity alone or *Depth* and *DSP* activities or *Depth, DSP* and *User* activities. During the *DSP* activity the DSP input is analyzed and three probable cell types are identified. Cell type and Layer type choices from the user are analyzed during the *User* activity. During *Infer Layer* the KBS looks at analyzed information from the *Depth, DSP* and *User* and determines a final layer type and cell type classification.

This concludes the process knowledge layout description of the Intermediate KBS. This is the level of proficiency that the KBS exhibits at the time of writing this thesis. The Intermediate KBS meets the design requirements, in that it is able to provide the user with a decision of where the probe is located within the three dimensional space of the brain. The Ultimate KBS of the future will contain additional features.

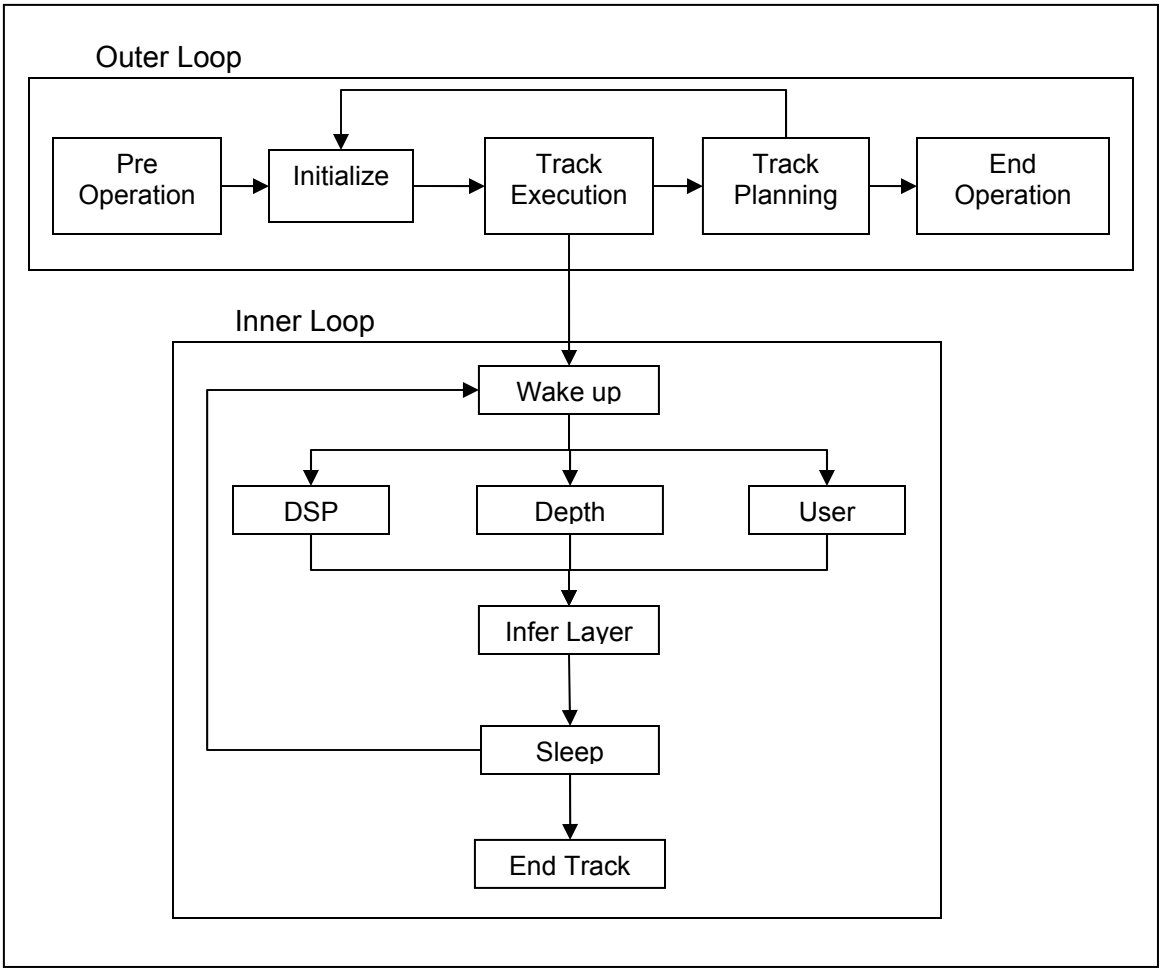


Figure 7.3. Intermediate KBS

7.3.3. Ultimate future KBS

The process knowledge layout of the two loops remains the same, as indicated by Figure 7.4. There are two additional activities namely the *History* activity and *Sensory-Motor* activity and one function namely the *Uncertainty* representation. Each of these constitutes possible future work.

1) The *History* activity.

In this activity each new event can be reasoned with and compared to information available from previous events, tracks or operations. Case-Based reasoning could be used to create this reasoning network.

2) The *Sensory-Motor* activity.

Create the ability to monitor body movement and compare this with the activity in the neuronal signal discharge. If the result is positive then the KBS should be able to identify what probable layer the probe is in. There is a correlation between motor activity, neuronal discharge and the area of the brain responsible for these reactions.

3) The *Uncertainty* representation.

All inputs are assumed to be certain in the Intermediate KBS. It could be beneficial to add tolerances, accounting for potential errors and uncertainties in the information received and produced by the KBS.

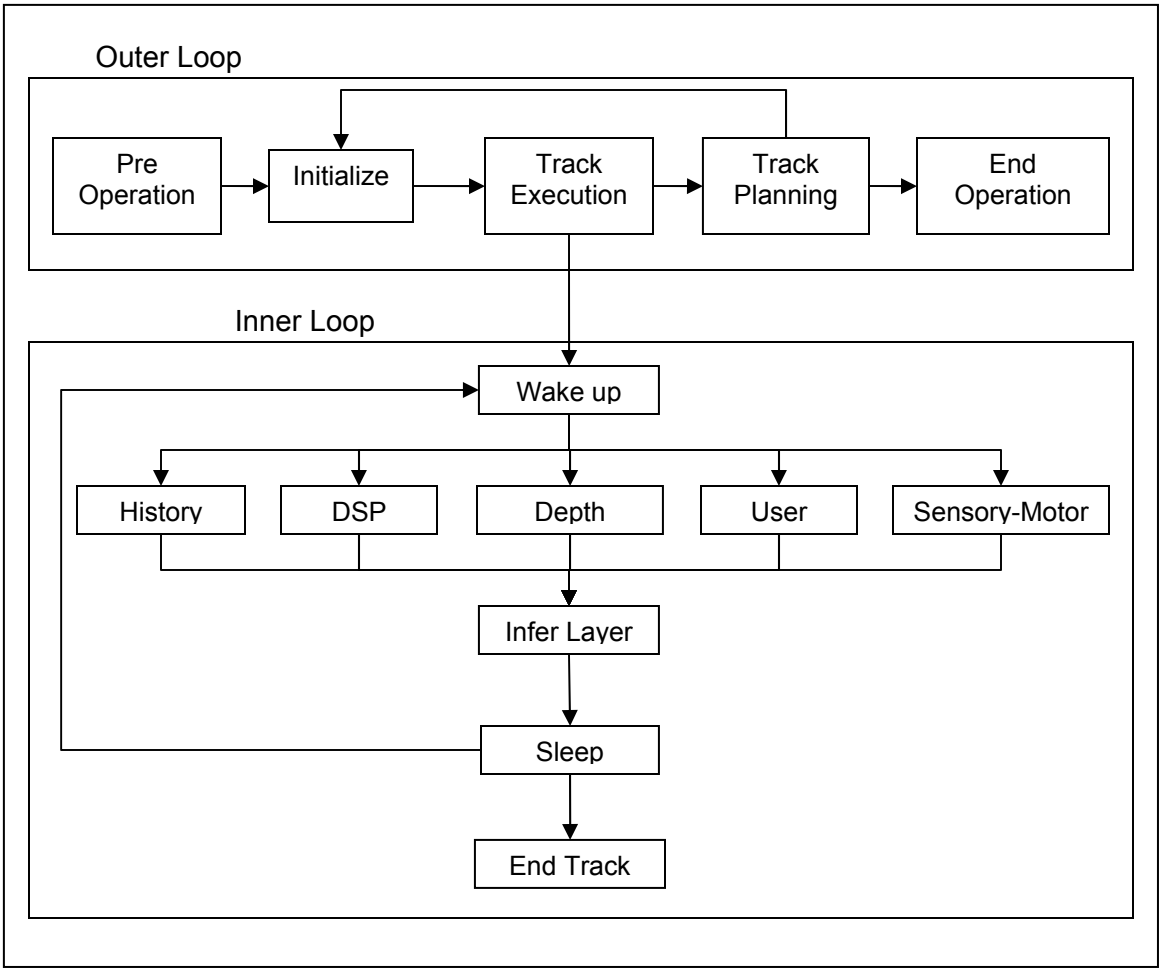


Figure 7.4. Ultimate KBS.

This completes the description of the Ultimate KBS. More functions could probably be added, but the Intermediate KBS already closely resembles the thought process of the expert, given the constraint of coupling the KBS with other Onetrack system components.

7.4. Implementation Description

At the time of writing this thesis the KBS refers to the Intermediate KBS presented in the previous section. The Intermediate KBS already closely resembles the thought process of the expert, given the constraint of coupling the KBS with other Onetrack system components. The knowledge engineer believes that the KBS meets all the design requirements set forth in the scope and therefore no further development is necessary. This section describes the different activities and the importance of each activity of the Intermediate KBS. The process knowledge and domain knowledge for each activity will be explained when applicable. The process knowledge deals with the order in which activities occur as shown in Fig 7.3. The process knowledge consists of an outer loop and an inner loop. The outer loop contains the activities that the neurologist is involved with before and after track execution. The inner loop is all the activities involved with the track execution. The knowledge encoded in the outer and inner loop is discussed in this section. The domain knowledge is responsible for reasoning with the incoming data and producing the intermediate reasoning steps. Where process knowledge deals with the sequence in which the activities are taking place, and how the results from different activities are combined to determine the final result, domain knowledge deals with the actual knowledge used to make the decisions. In most cases these are hard facts that are encoded and that play a significant role in the reasoning of the KBS. Throughout

this section generic examples, based on the equipment used at Emory University, are used to illustrate how the KBS works.

7.4.1. Outer Loop

As stated previously, the outer loop includes the activities that occur before and after track execution. This includes *pre-operation*, *initialize, track execution, track planning* and *end operation*. Each of these may be described as follows:

a) Pre-Operation

When the Onetrack system starts, the user is asked to input some general information about the patient and the operation. The information is sent to the KBS includes the patient's name, date of the operation name of the doctor and the side of the brain being operated on. The KBS uses this information to uniquely identify this patient's procedure. After the general information is entered the path plan must be initialized.

b) Initialize

The user is always responsible for entering the path trajectory into the Onetrack system. The 3D model takes this information and generates what is called a path plan in the KBS. The path plan indicates when the KBS should expect a new layer to begin. An example of the information contained in the path plan is presented in Table 7.1. The path plan consists of the predicted start and end depth of each anatomical layer. Figure 2.1. illustrates the anatomical layers.

Table 7.1. Example of a KBS path plan.

Layer	Start depth	End depth
Striatum (ST)	50	59
Lamina Exterior (Le)	59	61
Globus Pallidus Exterior (GPe)	61	67
Lamina Interior (Li)	67	69
Globus Pallidus Interior (GPi)	69	78
Lamina Anterior (La)	78	82
Internal Capsule (IC)	82	

Once the KBS has the path plan, it calculates the depths at which it should be awakened. The KBS needs to be awakened by this depth limit when the probe is:

1) at the beginning of a layer
2) half way through the layer
3) one millimeter before the end of the layer.

After the depth limit wakeup is calculated, the KBS is ready to commence with the Track Execution.

c) Track Execution

The knowledge and decisions involved in Track Execution are explained in the inner loop section 7.4.2. The primary objective of the Track Execution is to identify the cell and layer type of the probe tip at a given depth and point in time. During Track Execution the KBS notes at which depths the actual layers are found, this then generates the new (revised) path plan. The new path plan is used during Track Planning.

d) Track Planning

The KBS track planning strategy is based upon the expert's actions in the operation room. During the course of the procedure multiple tracks are executed, in order to verify the anatomy of the patient and to determine the best possible location to place the lesion or DBS lead. The KBS aids the user by suggesting in which direction to move the probe, after a track has been completed based on the new path plan. First the KBS tries to determine the anterior, posterior, medial or lateral region of the new path plan with respect to the Basal Ganglia. Then based on the information from the new path plan the KBS suggests the general direction in which the probe should be moved. The thickness of each primary layer in the path plan identifies a specific region in the brain. For ST, GPe, GPi, OT and IC a region of the brain, anterior, posterior, medial, lateral, central, positive or negative is assigned to each respective primary layer depending on the layer thickness (t). A summary of these decisions is given in Table 7.2.

Table 7.2. Corresponding layer thickness and orientation relationships.

Layer	Layer Thickness	Probable Orientation relative to Basal Ganglia
ST	t > 8	Anterior or Lateral
	t <= 8	Posterior or Medial
	None	None
GPe	t > 6	Anterior
	4 <= t <= 6	Central
	t < 4	None
GPi	t <= 4	Anterior, Posterior or Lateral
	4 < t < 6	Medial
	t >= 6	Central
	None	None
OT	Found	Positive
	Not found	Negative
IC	Found	Positive

	Not found	Negative

For ST, GPe and GPi the probable orientation is based strictly on the thickness of each respective layer. For OT and IC the thickness is irrelevant, but what is important is that the layer was found or not found. If it is found it is assigned positive, otherwise negative. The process knowledge uses this information to determine the ultimate result. Table 7.3. is an example of how the thickness of each layer is translated into a probable orientation relative to the Basal Ganglia.

Table 7.3. Example of determining probable probe orientation.

Layer	**Layer Thickness (t)**	**Probable orientation**
ST	10	Anterior or Lateral
GPe	5.8	Central
GPi	3.8	Anterior, Posterior or Lateral
OT	Not found	Negative
IC	Not found	Negative

Certain combinations establish where the next track should be placed. These combinations are summarized in Table 7.4.

Table 7.4. Track Planning layer combinations and results.

Primary Layer					Result
Striatum	**GPe**	**GPi**	**OT**	**IC**	
None	Any Answer	Any Answer	Negative	Negative	Check Equipment
Any Answer	None	Any Answer	Negative	Negative	Check Equipment
Any Answer	Any Answer	Any Answer	Any Answer	Positive	Move 2mm medial or lateral.
Any Answer	Any Answer	Any Answer	Positive	Positive	Move track medial, lateral or posterior
Any Answer	Any Answer	Any Answer	Positive	Negative	Move 2mm posterior
Any Answer	Central	Central	Negative	Negative	Move 2mm posterior
Anterior	Anterior	Anterior	Negative	Negative	Move 3-4mm posterior
Anterior	Anterior	Any Answer	Negative	Negative	Move 3-4mm posterior
Anterior	Any Answer	Anterior	Negative	Negative	Move 3-4mm posterior
Not Anterior	Anterior	Anterior	Negative	Negative	Move 3-4mm posterior
Lateral	Any Answer	Lateral	Negative	Negative	Move anterior, medial or posterior
Posterior	Any Answer	Posterior	Negative	Negative	Move 1mm posterior
Medial	Any Answer	Medial	Negative	Negative	Move 2mm lateral or stop procedure.
Not equal to GPe or GPi	Not equal to Striatum or GPi	Not equal to Striatum or GPe	Negative	Negative	No answer

The knowledge statements represented here are taken directly from the knowledge acquisition. For example, *None* indicates that this particular layer was not found. If for any reason the Striatum or GPe is not found, then it is necessary to check the equipment and the results from the track are discarded. *Positive* indicates that the Optic Track (OT) and/or Internal Capsule (IC) is found. *Negative* means they are not found. The results are only suggestions of where the probe should be moved to for the next track execution. It is entirely up to the user to ultimately decide where the next track should be placed. Given the example in Table 7.3., the suggestion of where the probe should move to is either *Move 3-4mm posterior* or *Move anterior, medial or posterior* when examining Table 7.4. At this stage it is up to the inference engine to determine one or the other selection.

Once the Track Planning is completed the user can either start a new track or end the operation. In case of a new track the entire process starts over with the Initialize activity.

e) End Operation

The KBS is notified once the user decides to end the operation. Note that the user is able to end the operation at any time during the KBS process. Once the KBS is notified that the operation is completed, it will proceed to dump all of the reasoning context into a text file that can be viewed by the user at a later stage.

7.4.2. Inner Loop

The primary purpose of the inner loop is to identify the layer and cell at the probe tip at a given depth and time. It is of utmost importance to the user to know where in the brain the probe is, for safety and the mapping of the structures. The inner loop stems from the Track Execution of the outer loop. The activities (described below) involved with the inner loop are described below and are *Wake up, DSP, Depth, User, Infer Layer, Sleep* and *End Track*.

a) Wake up

The KBS is awakened whenever an event occurs during track execution. Recall that an event is defined as when the user inputs information, the DSP generates a cell type probability array or a depth limit is reached, as discussed in the Scope (Section 7.2.). The KBS is awakened whenever the user selects a cell and or layer type for the probe at a given depth at a certain point in time. The DSP is capable of awakening the KBS when it determines a probability array of cell types. Lastly the KBS sets specific depths at which it has to be awakened as discussed in Section 7.4.1.b. It is however not necessary for all of these sources of data to be available to the KBS at the same time. The actual mechanism used to wake the KBS up is described in Chapter 8.

b) DSP

The DSP produces a probability array of cell types that are analyzed by the KBS. Before describing this process there is an important list of layer and cell type relations that need to be explained. These relationships play a vital role in the KBS reasoning capabilities. Table 7.5 indicates the layer-cell type relationships.

Table 7.5. Layer-Cell type relations.

Layer Type	Cell Type
ST	Caudate, Popcorn and Border
Le	Fiber and Border
GPe	HFD-P, LFD-B and Border
Li	Fiber and Border
GPi	Tonic, Bursting, Chugging, Pausing, Tremor and Border
La	Fiber and Border
IC	Multiple, Injury and Neg
OT	None
Any	Artificial
None	None

Each layer is associated with a specific set of cells. This knowledge was obtained from the initial knowledge acquisition, and is one of the key reasoning components within the KBS. It is used extensively to determine both the cell and layer classifications. The cell "names" or types are based on the sound that the neuronal discharge makes when encountered by the probe tip. This leads to names such as popcorn or burster. A popcorn cell sounds like popcorn popping in a microwave and a burster cell sound like short intermittent bursts.

The DSP provides the KBS with a probability array of cell types corresponding to those in Table 7.5, as well as the value for the signal-to-noise ratio. The procedure is illustrated in Figure 7.5. First the KBS decides whether this is a primary or secondary layer-cell type relation based on the signal/noise ratio of the probe. If the signal/noise ratio is greater than 15 it indicates a primary layer, otherwise a secondary layer is indicated. The value 15 was obtained through knowledge acquisition. The KBS then examines the probability array of the DSP and selects the highest three cell types to continue reasoning. If the three cell types are fiber, border

and HFD-P and the signal/noise ratio is 20, then the KBS will identify this as a primary layer. The KBS then looks at the three selections and only allows those falling in a primary category (ST, GPe, GPi, IC) to pass through to the next stage of reasoning. The other cell types are discarded. Hence in this example only the border and HFD-P cell types will be assigned as probable answers for the DSP source, based on Table 7.5. The only exception to this rule is when the maximum selected cell type from the probability array is an artificial cell, indicating that the probe picked up some artificial noise. The user is then notified that there is too much artificial noise and that the DSP can not detect the desired cell types. Artificial noise occurs when the probe picks up sounds that are not related to the neuronal discharge.

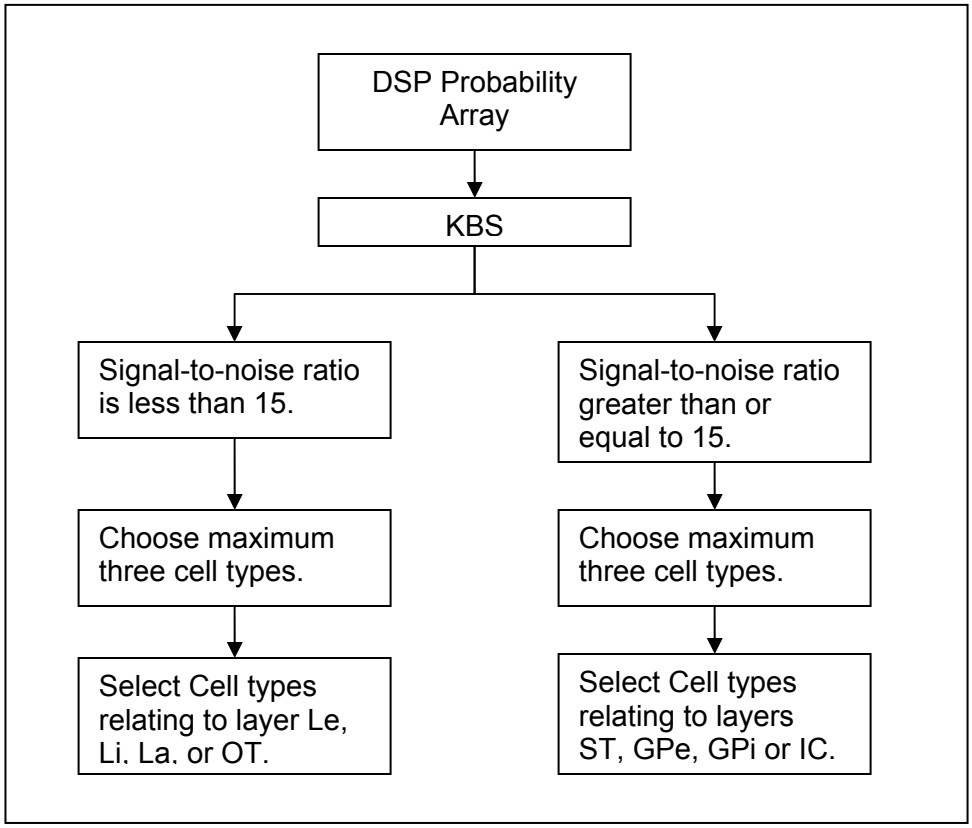


Figure 7.5. Diagram flow for selecting DSP cell types.

A strength of the KBS is that a cell type can be assigned independently of the user. If the DSP detects a prominent cell type it notifies the KBS. If the answer is acceptable within the path plan parameters then the cell type is assigned. However, a problem that arises is how frequently the DSP should send information to the KBS. The DSP is capable of producing a probable cell type array every second. This does not mean that the DSP finds a cell type every second, but instead that the neuronal discharge signal is analyzed every second. The KBS can not handle the information at such a rapid rate. Thus it is necessary to have a filter between the DSP and KBS. The KBS only receives the DSP array when the velocity of the probe is zero, or when a significant change in cell type occurs. If the same cell type is found for 20 seconds, then the KBS is only notified once. Thus the amount of information coming into the KBS is limited to significant events within the track as controlled by the user interface. Note that the DSP activity only assigns probable cell types for the probe tip at a given depth and point in time.

c) Depth

During this activity a probable solution for a cell and layer type is assigned based on the depth of the probe tip and the path plan. Assume that at some point during the track execution the depth of the probe indicates that the probe should be in the GPe, when compared to the path plan. Using Table 7.5., the KBS assigns HFD-P, LFD-B and Border as the probable cell type answers from the Anatomy source.

The following three steps are involved with the determination of the appropriate layer type from anatomy. The depth value of the probe tip and path plan are the only data points that are entered into the KBS during a track execution.

1) The current depth is compared directly to the path plan and depending on which boundary the current depth falls, a specific layer is assigned. The layer boundaries consist of a starting and ending depth. For example the starting depth for the ST is 50mm and its ending depth is 56mm, based upon the Emory surgical equipment. Therefore, if the current depth is 52mm then ST will be assigned as the probable layer type.
2) Next the KBS determines whether the probe moved up or down. The KBS then compares the current layer to the previous layer in order to ensure that an acceptable progression exists. For example, one can not have the GPi followed by the ST if the probe is moving downward based upon anatomical structures. Table 7.6 summarizes the allowed progression of layers as the probe moves downward or upward. Figure 2.1 illustrates the different anatomical layers and the orientation of these layers.
3) If the progression is acceptable then the probable layer is assigned as the answer for the Depth layer. If it is not acceptable, then the program will halt because an illegal operation was performed.

Table 7.6. Allowed progression of layers.

Probe moves down		Probe moves up	
Previous Layer	**Current Layer**	**Previous Layer**	**Current Layer**
None	ST, Le, GPe, Li, GPi, La.		
ST	ST, Le, GPe	ST	ST, None
Le	Le, GPe	Le	Le, ST
GPe	GPe, Li, GPi	GPe	GPe, Le, ST, None
Li	Li, GPi	Li	Li, GPe
GPi	GPi, La, OT, IC	GPi	GPi, Li, GPi
La	La, OT, IC	La	La, GPi
OT	OT, IC	OT	OT, La, GPi
IC	IC	IC	IC, OT, La, GPi

Note that the Depth activity assigns probable cell and layer types for the probe tip at a given depth and point in time.

d) User

The answer that corresponds to the User is entered directly by the user and is allowed to pass through to the next stage of reasoning without undergoing analysis (Figure 7.3). The cell and layer type selections are obtained from the user. The cell and layer type selections are not evaluated but allowed to pass through to the next reasoning phase. For the example, say that the user has selected HFD-P and GPe, and then this information is made available to the Infer Layer activity directly. Note that the User activity provides a probable answer for the cell and layer type classifications.

e) Infer Layer

Inferring the ultimate layer is a two step process. First of all it is necessary to identify the cell type that the probe tip is in at a given depth at a certain point in time. Then based on this conclusion and other sources of information the KBS is able to identify the layer. Following then is a description of identifying the cell type first, followed by a description of identifying the layer.

1. Identify Cell Type

Identifying the cell type at a specific depth is a crucial step in deciding in which anatomical layer the probe tip is located. Cell type classification is one of the primary pieces of information that the user relies on to determine the location of the probe. It is therefore important for the KBS to correctly reflect the process of classifying cell types. The KBS has three sources of information available when classifying the cell type as indicated in Figure 7.6.

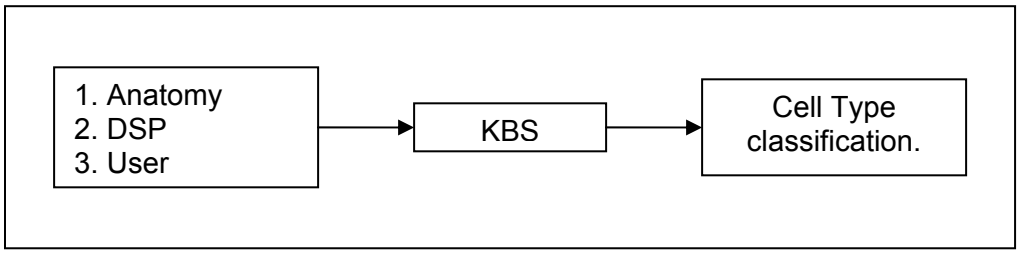


Figure 7.6. Determining factors involved with Cell Type classification.

Each of the three sources of information, Depth (h), DSP (d) and User (u) is capable of classifying a unique cell type. Thus it becomes necessary to combine the information from the three sources and determine one solution. A set of

process knowledge statements (summarized in Table 7.7) that deals with combining the probable cell types and classifying the final cell type.

Table 7.7. The cell type classification combination rules.

Cases	**All sources of information available**	
	Combination	**Solution**
1	h=d=u	u
2	h=d but d≠u and h≠u	Question the user.
3	h=u but u≠d and h≠d	u
4	d=u but h≠d and h≠u	u
5	h≠d≠u	u
	Only Anatomy and DSP sources available	
	Combination	**Solution**
6	$h=d_1$	d_1
7	$h=d_2$	None
8	$h=d_3$	None
9	h≠d	None
	No Anatomy source available	
	Combination	**Solution**
10	h=none	None
11	h=none but d=u	u
12	h=none and d≠u	u

The ideal situation is when the answer from Anatomy, DSP and User (h=d=u) match. This is the governing rule and is always considered first. In the event that the answers do not match the other combinations are tested. If information from all the sources are available, then by default the user's answer is selected. However in the event where Anatomy and DSP are the same but disagree with the User's answer (h=d but d≠u and h≠u), then the user is asked to reselect the answer. If the only sources of information available are Anatomy and DSP, then the only acceptable answer is when Anatomy matches the maximum (d_1) cell type from the DSP array of probable cell types ($h=d_1$). Otherwise, no cell type is

assigned. If the only source is Anatomy, then no cell type is assigned. The only alternate situation that arises is when Anatomy has no answers. The user's answer is assigned by default.

Once the final decision for cell type is made it can be converted to a layer type by using Table 7.5.

2. Identify Layer Type

Typically the user identifies the cell type first and then infers a layer type based on the cell type. This is also the case in the KBS. Based on domain knowledge, the KBS is capable of inferring what layer the probe tip is in, based on the assigned cell type only. However, in the KBS there are other sources of information available that the user does not have, and thus it is necessary to combine these answers.

As mentioned previously, there are three sources that play a role in classifying the layer type, Depth (h), Cell type (c) from previous section and the User (u) selection as illustrated in Figure 7.7.

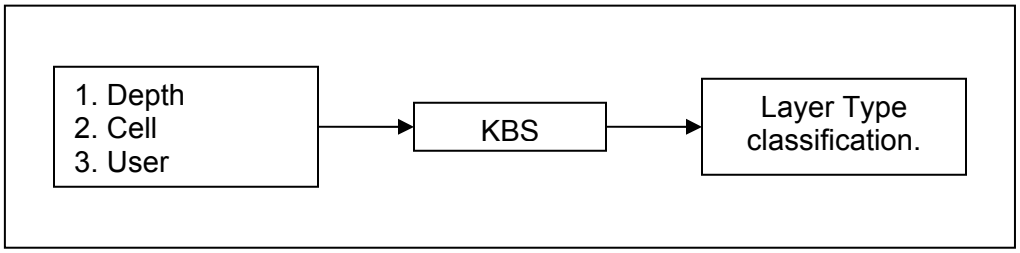


Figure 7.7. Determining factors involved with Layer Type classification.

Each of the three sources are capable of determining a layer type based on domain knowledge. It is necessary to have a method of integrating the individual answers and come to a conclusion as to which layer the probe is in. The rules illustrated in Table 7.8. are used to combine these answers and classify the layer.

Table 7.8. The layer type classification combination rules.

Case	**All sources available**	
	Combination	**Solution**
1	h=c=u	u
2	h=u but h≠c and u≠c	u
3	u=c but c≠h and u≠h	u
4	h=c but c≠u and h≠u	Question the user
5	h≠c≠u	u
	Only Depth and Cell available	
	Combination	**Solution**
6	h=c	c
7	h≠c	c
	Only Depth available	
	Combination	**Solution**
	h	h
	Other	
	Combination	**Solution**
8	No h but c=u	u
9	No h but c≠u	u
10	No h, no u but c	none

The ideal situation would be when the Depth (h), Cell classification (c) and user (u) answers match (h=c=u). This is the governing rule. Only when this is not the case are the other combinations considered. In the case where all sources are available the user's answer is used by default. The exception to this is when the Depth and Cell classification match, but disagree with the use, in which case the user is prompted to reselect the layer type (h=c but h≠u and c≠u). A strength of this approach is that a layer type can be identified without input from the user.

By default the KBS will always identify the layer that corresponds to the path plan at a specific depth, especially when no other information but depth is available. In the case where only depth and cell classification information is available, a layer is assigned when the answers match. Otherwise nothing is assigned.

The final layer type classification is thus based on more than just Cell type classification. The KBS takes into consideration the path plan constraints as well as the user input. As the track progresses, the KBS generates a new path plan that reflects the actual path that was identified. If all the input sources are available and they match, then the layer is classified as a solid layer. Otherwise the layer is classified as fuzzy, which is indicated to the user when the KBS completes the reasoning for that specific event. At the conclusion of the track all sites found are used to determine where the track is located in three dimensional space of the brain.

Once the layer and cell is identified, this information is made available to the user.

f) Sleep

During the Sleep activity the KBS merely waits on the other components of the Onetrack system. The layer and cell identifications are passed on to the user and the KBS stops processing. The KBS can either be awakened again or the user can end the track. If the KBS is awakened this process starts over at the Wake Up activity.

g) End track

The user can select to end the track at any stage of the operation. However, it should be noted that once the user has ended the track the KBS can not go back to reasoning with that track information. Once the user selects end track the KBS completes the Track Execution activity and proceeds with Track Planning.

7.4.3. Summary of Process and Domain Knowledge

The process knowledge is the back bone of the KBS in that it regulates all the reasoning activities that take place. It is intricate and complicated, but ultimately comes together to provide a solution. The process knowledge of the KBS consists of an outer loop and an inner loop. The outer loop consists of everything that takes place before and after the track execution, while the inner loop deals solely with track execution. The Intermediate KBS takes into consideration information available from the 3D model, DSP and user. By considering the path trajectory (generated by the 3D model), depth reading of the probe, DSP probable cell type array and user input, the KBS is able to sift through all the data and come up with a final answer. This final answer contains the final classification of the layer type and cell type at a specific depth at a specific point in time. After completing track execution the KBS will also propose where the next track should be placed. Domain knowledge can be very illusive at times. However the best way to think of it is as that part of the KBS containing all the intricate details of reasoning. The domain knowledge are statements that will not change over time, and is therefore reliable. In some instances the domain knowledge is supported by supplemental information that is currently not available to the user and thus the answers are interpolated to best fit the situation.

7.5 Code

The code is written in the CLIPS platform [29]. This software is freely available and can be obtained by downloading the software from the internet. The actual KBS code format follows that which is described in Giarratano's book "Expert System Principles and Programming" [27]. A hard copy of the KBS code appears in Appendix D. A user's manual for running the KBS in CLIPS appears in Appendix E. Appendix F gives an explanation for every rule, template and fact used in the KBS.

7.6 Summary

Process knowledge deals with the order in which the reasoning activities take place. Domain knowledge contains the detailed facts and functions that the activities use to produce a valuable answer. The scope of the KBS is to classify the layer and cell type of the probe at a specific depth at a given point in time. The KBS developed here exhibits the intermediate KBS process knowledge layout. It primarily uses information from the path plan, current depth, DSP and the user to suggest where the next track should be placed. However, the final decision and responsibility of cell type classification and layer determination is the users.

CHAPTER 8

COMMUNICATION PROTOCOLS

8.1 Introduction

The KBS and main program of the Onetrack system are written in two different platforms, thus to share data it is necessary to create a link between them. A communication protocol is therefore created to serve as the communication link between the Knowledge Based System and the main program of the Onetrack system. A unique set of language classifiers are developed for the input and output of the KBS to or from the Onetrack system components. An activation mechanism, using ActiveX control, is created so that the KBS can execute whenever applicable.

8.2 Need for System Interaction

An interface is designed between the KBS and the user interface (UI) components of the Onetrack system. This is necessary because each system component operates within an environment specific to that component, shown in Figure 3.1. The interface is accomplished through the creation of a specific communication protocol. The protocol lays down a common method of how information is to be sent from one component to the other.

The KBS and the Onetrack system needs to work in harmony to accomplish tasks set forth by each. The user must be able to query the KBS for a decision on probe location, given a certain set of parameters by the user and the other components of the Onetrack system. In turn, the KBS must be able to request certain activities to be performed by the Onetrack system in order to provide the necessary guidance for the user. These

activities include questioning the user, or presenting data to the user, or even requesting updated information from the DSP or 3D modeling components. After the KBS produces a result, all control of the Onetrack system is returned to the user.

Several assumptions are made in creating this interface.

1. The user is capable of proceeding in the absence of data and information. Thus the Onetrack system has to be able to also reason in the absence of information and data, therefore the need for a KBS.
2. The user is in control of the Onetrack system and makes all the final decisions concerning the track and surgical procedures.
3. The user is able to query the KBS and the KBS is able to query the different Onetrack system components to aid in the decision making process.
4. There is a direct link between the User Interface (UI) and the KBS in that all data into the KBS goes from or through the UI and all KBS decision outputs or queries go directly to or through the UI.
5. The KBS can be activated by the UI when certain circumstances exist and be asked to do some reasoning. At all other times the KBS will not be running.
6. A language is created to serve as the common method of communication. This language is universal throughout the Onetrack system and can be implemented in various ways as long as the respective environments are able to interpret the symbols.
7. The KBS is to be programmed using CLIPS [29] and the UI using C++.

The following section contains a detailed description of the language that makes up the communication protocol.

8.3 Language Description

The language serves as a communication link between the KBS and the UI. By agreeing on set usage and meanings of symbols, the system is able to interpret output to/from the KBS. This is a two way communication system between the KBS and UI. The KBS needs to send information to the UI and the UI needs to communicate with the KBS. Therefore there are two parts to the language description. First is the input that the UI sends to the KBS and secondly the output from the KBS to the UI. It is impossible to have one without the other.

8.3.1. Interface from UI to KBS

There are two primary reasons for the UI to send information to the KBS. The first is that the UI is responding to information presented by the KBS. Secondly, the UI sends information to the KBS when an event occurs. Recall from Chapter 7 that there are three occurrences that constitute an event; namely, current depth information is available, current depth and DSP information is available, or current depth, DSP and user information is available. Asserting information into the KBS is a two step process. First the KBS is told to get ready for new information and then secondly the information is sent over to the KBS.

The environment in which the KBS will run is CLIPS. The Onetrack system “asserts” information into CLIPS by adding statements in appropriate KBS, syntax to the list of facts in CLIPS. A fact contains the data on which inferences are derived. First it is necessary to assert information to notify CLIPS whether this is a UI response to information or event information. When the UI is responding to information presented by the KBS the first assert appears as follows:

(assert (goal answer status available)).

If an event occurs, the following statement is first asserted into the KBS

(assert (goal name kbs status wakeup)).

Note that this last assert triggers the inner loop of the KBS process knowledge to execute. In affect this statement “wakes” the KBS up and activates the KBS to reason within the track execution parameters of the code.

The second assert sends the actual information over to the KBS. There is a set format that is always used to send information over to the KBS. The second assert uses the following syntax format, which is described in detail in Table 8.1.

(answer (yn-answer ?a)(multichoice ?b)(timetag ?c)(current-depth ?d)

(text-field ?e)(pathplan ?f)(dsp-cell ?g)(format ?h)(status ?i)(user ?j))

Table 8.1. Syntax variable used in the UI to KBS communication.

Field		Description	Function
1.	answer.	The name of the template.	Internal KBS usage for interpreting.
2.	yn-answer.	Contains a yes or no answer to a question posed by the KBS.	The KBS poses a question to the user to obtain a positive or negative response to the existing circumstances.
3.	multichoice.	This field contains the answer selected by the user.	The KBS poses a question to the user that is made up of a list of variables from which the user must select one.
4.	timetag.	An integer value is assigned by the Onetrack system and increases each time a new assert is made.	Used as a reference point so that the KBS knows that it is receiving new information.
5.	current-depth.	Current depth reading of the probe tip obtained by the equipment.	Used to determine the location of the probe relative to the desired path trajectory. This information will always be available.
6.	text-field.	Any typed text response by the user.	For when the user wants to make notes concerning a certain decision.
7.	pathplan.	Starting and ending depths for each anatomical layer provided by 3D modeling.	Used as frame of reference against which all layer classification is tested. Also known as the path trajectory.
8.	dsp-cell	A list of probabilities associated with cell type classification as determined by digital signal processing.	Acts as another form of verification that the probe tip is where it is expected to be.
9.	format	Indicates where the information is from, whether it is from model, user, etc.	Internal KBS usage for interpreting.
10.	status	Indicates whether the information is received by the UI or not.	Used only to notify the KBS that a message was received. No data is sent back to the KBS in case of a message only.
11.	user	Contains user specific answers for an event.	The KBS uses the user event information to reason with.

Note that the fields that are specifically associated with an event are current-depth, dsp-cell and user. The fields involved with responding to a question or request are yn-answer, multichoice, path plan and text-field. There are common fields that are always passed to the KBS, regardless of the circumstances. These fields are timetag, format and status.

There are three fields that are further defined for the purpose of obtaining the information in the correct format. These three fields are path plan, dsp-cell and user. Each of these three are explained further in the following examples.

Example A.1.

The KBS sends a request to the Onetrack system to supply it with a path plan, before track execution. The predicted path trajectory comes from the 3D model and consists of the starting depth for each layer. The path trajectory is converted into the path plan when entered in the following way

(assert (goal answer status available))

(assert (answer (timetag 3)(pathplan ST 43 Le 53.85 GPe 56.84 Li 63.61 GPi 64.43 La 68.21 OT 75.7 IC 76.92)(format model)))

Note that two asserts are done. The first notifies the KBS that information is available and the second sends the path plan over. This exact format should be followed in order for the KBS to accept the path plan. Note that the KBS format is case sensitive. The path plan consists of telling the user what the layer name is and then the depth at which that layer is expected to start and end. For example, ST 43 means that when the probe gets to a depth of 43 then it will be in the Striatum layer for the first time. The values for the depth are based upon the surgical equipment

used by the Emory University hospital. All components of the Onetrack system use this and are calibrated.

Example A.2.

The DSP detects a significant cell type and sends this information over to the KBS. The following format is used for such an event

(assert (goal name kbs status wakeup))

(assert (answer (timetag 17)(current-depth 62)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 50 LFD-B 20 border 30 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)
(format dsp)))

Two asserts are done. The first indicates that this is event information and that the KBS needs to "wake up". Secondly the information is sent over to the KBS. The probability array for dsp-cell is generated by the DSP Onetrack system component. The fields for the dsp-cell are *sn_ratio, background, caudate, popcorn, HFD-P, LFD-B, border, tonic, bursting, chugging, pausing, fiber, multiple, artificial, injury, tremor* and *neg*. The format field indicates that this information comes from the dsp. Thus the user has no involvement with the information being sent to the KBS, it is all automated.

Example A.3.

The user, DSP and current depth information all become available to the KBS. The following syntax is followed to send information to the KBS

(assert (goal name kbs status wakeup))

(assert (answer (timetag 14)(current-depth 60.3)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 50 border 10 tonic 40 bursting 10 chugging 60 pausing 10 fiber 20 multiple 10 artificial 13 injury 10 tremor 10 neg 10)
(user layer ST cell popcorn)(format user)))

This is an event wakeup created by the user. The only acceptable parameters for the user input are *layer* and *cell*. Note that the only acceptable values for *cell* are those outlined by the DSP (Table 7.4) and the values for layer are those outlined by the path plan. The *format* value indicates that the information comes from the user. Whenever the user creates an event, the *current-depth* and *dsp-cell* fields need to be completed as well.

8.3.2. Interface from KBS to UI

The purpose of the KBS is to provide the user with the classification of layer and cell type at a specific depth. This set of information is sent to the user only at the end of *track execution*. A new event can only be created when the KBS has completed the reasoning for the current event. In some cases the KBS might need more information before a final decision can be made, and it can request this from other Onetrack system components or directly from the user. This is done by posting a statement to the Onetrack system and then halting the KBS. The KBS continues once the necessary information is received back from the Onetrack system. The information is encoded using a specific language format that allows easy parsing and understanding for the UI. The following syntax is used

Type:Format*Variable*Text_field!

The string consists of any number of commands from the KBS delineated by the “!” character. Each command is a set type that is indicated by a one letter code, as described in Table 8.2.

Table 8.2. Syntax variable used in the KBS to UI communication.

Field		Description	Function
1.	Type.	The KBS specifies the action that needs to be taken by the system. There are four types of responses that the KBS may give, these are questions (Q), request (R), message(M), and data (D).	In certain instances within the KBS reasoning, a specific task needs to be completed by the Onetrack system to produce a desired answer. Q is used when posing a question to the user, R when requesting data from the system, M when displaying a message to the user, and D when providing the final decision to the system.
2.	Format.	This is a specification field in order to be more descriptive of what Type is.	Format specifies what action or display is associated with type. In some cases this field is the same as the format field that is part of the information coming into the KBS.
3.	Variable.	This is a classification of what variables are involved in the reasoning.	Places a limitation of the response that the KBS is looking for.
4.	Text field.	Text description associated with the action that is required.	Used especially for communication between the KBS and the user.

Below are three examples of responses that might be produced by the KBS and a description of when these responses may occur.

Example B.1.

M:message*The KBS has been activated and is standing by.!

The KBS needs to display a message stating that the KBS is initialized and ready to start the procedure. This is the first message that will always be displayed when the KBS is activated. The Type is M for message, the format is message, no variable is necessary in this example, and the text field is “The KBS has been activated and is standing by.” This message appears on the screen for the user to see. The rest of the syntax, such as the colon and the asterisk, is used by the C++ user interface and is not displayed. These characters act as separators between fields. The exclamation mark at the end signals the end of the string being passed to the UI

Example B.2.

R:model*first pathplan*!

The KBS is requesting the path trajectory necessary for track execution. The path plan is entered by the Onetrack system as shown in Example A.1. Type is R, the format is model because the information is originating from the 3D model component of the Onetrack system. The variable is “first pathplan”. This lets the model know that the KBS is seeking the predicted path trajectory.

Example B.3.

D:wakeup*(65.12)*(nil)!

D:data*(Layer Le Cell none State fuzz)*(nil)!

This is what the KBS sends to the Onetrack system to notify it of the final decision for classifying the layer type and cell type at a specific depth. The first statement has wakeup in the format field; this indicates that the maximum depth the probe can reach without awakening the KBS. In this example the probe can not go deeper than 65.12 without sending updated information to the KBS. These values are calculated from the path plan and are updated as the track progresses. The second statement contains the decision for layer and cell type for the current event for which the KBS was sent information. The type is D for data in both cases. The format for the second is data. The variables are the data sets that are being sent over to the Onetrack system. Note that the text field contains *nil*. *'nil'* is always used whenever a field is inactive, this is the default setting.

The communication protocol is complex and is best understood by closely examining the examples. For a detailed understanding of the interface it would be best to run the KBS in CLIPS and experiment with the various input fields. Appendix E contains a user manual for running the KBS in CLIPS.

8.4. Communication Mechanism

An important task in the development of the Onetrack system is the integration of the main software written in C++ with the KBS, which uses the CLIPS engine. A CLIPS ActiveX control was found that encapsulated the CLIPS engine, which allows the C++ program to run CLIPS files, assert facts, etc. [30.] Having this ActiveX control means

that the KBS can become part of the Onetrack system and does not require running any other programs in parallel.

However, the way a CLIPS file has to be written for use via the ActiveX control, is different from the way it might be traditionally written for use in a stand-alone CLIPS environment. This meant that the CLIPS developer in this project was initially using user prompts to request information for the KBS when running it in the CLIPS environment. This process is different from the Onetrack system in which information is passed programmatically to CLIPS via an “assert”. Differences such as this meant that the developer would often write code for use in the CLIPS environment for ease of testing and then later adapt it to work in the C++ program. The integration of the main application and the KBS required that the CLIPS developer gain an understanding of C++ conventions. On the other hand, the C++ developer had to acquire a basic understanding of CLIPS in order to interface correctly with the ActiveX control.

The CLIPS ActiveX control allows the main program of the Onetrack system to make calls to the CLIPS engine using standard C++ function calls. This includes the ability to load CLIPS files, assert facts, run CLIPS programs, and to view the current state of the KBS. The ability of the main program to assert facts to the CLIPS engine provides a way to pass information to the KBS. In the other direction, the CLIPS ActiveX control provides a way for the KBS to write a string containing data that can be accessed by the C++ program.

The information is passed to the main program via a string and the user’s answers are passed back via asserts. Typically the CLIPS engine is running continuously. In the

Onetrack system the C++ “awakens” the engine by calling a “run”; CLIPS then executes its reasoning until there is nothing else to do with the available information and then halts. It is the responsibility of the main program to alert CLIPS/KBS when relevant information is made available and then call “run”.

To implement CLIPS communication, a C++ class was created. The class provides code for initializing CLIPS as well as methods that are called when pertinent events occur. Each event results in two assertion strings in a queue. Meanwhile, a thread in the program is working through the queue. This thread pulls the next assertion string from the queue and the assertion is sent to CLIPS. Then the thread calls “run” and waits for the call to return. If there is a response from the KBS requesting more information then it is dealt with first. Those responses may result in more assert strings, which are then asserted and CLIPS is “run” again. Only when the final decision concerning the layer and cell type classification comes from the KBS, does the thread create a new event. In this manner, order and synchronization can be maintained (i.e., a new cell that has been identified does not interrupt the analysis of a previous cell that is being identified by the KBS).

On the KBS side the rules used with the main program are set outside the process knowledge of the KBS. The KBS output rule can be fired and information sent to the main program at any time within the KBS code structure. The output rule will halt the KBS inference engine until new information is received. This rule is referred to as the “sleep mechanism.” Only the correct assertions from the Onetrack system can wake the KBS and cause it to continue the execution process.

8.5. Summary

The KBS and UI interface communication is accomplished by using a common language that can be interpreted by each program. This is a two way communication system. The UI communicates to the KBS by asserting facts via the CLIPS ActiveX control. It is important that the exact syntax is used because the KBS is case sensitive. There are two primary reasons to send information to the KBS. First, when the UI is responding to a request issued by the KBS. Secondly information is sent whenever an event occurs. The KBS sends information to the UI by posting a string to the buffer that is read by the main program. This communication protocol makes it possible for the KBS to be an integrated component of the Onetrack system.

CHAPTER 9

KBS EVALUATION

9.1 Introduction

The KBS is tested in order to determine the reliability of the computer program with respect to the knowledge that it represents. The KBS is compared to the knowledge that is gathered during Knowledge Acquisition (KA). Secondly the KBS is tested extensively in the CLIPS environment. Both of these tests are done on a continuous basis from moment to moment as the KBS procedure progresses.

9.2 Procedure for Testing KBS

There are four methods for testing the KBS. First, the knowledge in the KBS is compared to the Knowledge Acquisition findings. The second method involves testing the KBS in the CLIPS environment; using the CLIPS environment makes the debugging process easier because the facts and decisions can be observed as they are generated in real time. Thirdly, the KBS should be tested in the integrated Onetrack system. At this stage of the development the Onetrack system has not been completed, so all the necessary tests can not be performed in this environment. Finally the entire Onetrack system will be tested in the operating room to determine the efficiency and correctness of the entire Onetrack system. Based on the recommendations from the experts testing the Onetrack system, it might be necessary to revise and improve the entire Onetrack system.

The knowledge engineer performed the first two test procedures, namely Knowledge Acquisition testing and the CLIPS environment testing. Additionally, limited test cases were performed with the Onetrack System.

9.3 Knowledge Acquisition Testing

The knowledge obtained during KA is incorporated into the KBS. Therefore, this test determines how closely the KBS resembles the knowledge that it represents. This is done by examining the main points gathered through Protocol Analysis to the encoded knowledge of the KBS. This is a very objective method of testing. If an important section of the knowledge is missing, then the KBS needs to be redesigned before commencing with the next test phase.

This testing is done throughout the code development of the KBS. As each new functionality is added, it is compared to the gathered knowledge. There are several facts that are encoded into the KBS that come directly from Knowledge Acquisition. Some of these are the cell to layer type combinations, the order in which layers are found in the brain, and how to determine where to place the next track. Some rules are more intuitive as they relate directly to the new technology that is available. For example, the availability of a cell type probability array from the DSP. In the past this has not been available to the user. Therefore most of the reasoning knowledge is obtained from examining the data source and logically determining how to interpret the data. Some of the functions depend on the user's data input which has the highest validity. This allows the user to override answers produced by the KBS, which always leaves the user in control.

It is the belief of the knowledge engineer that the knowledge represented in the KBS is reflective of the KA, given the design constraints. Only after the Onetrack system is used in the field will it become apparent what affect the system has on the methodology of the procedure. Therefore a redesign is eminent, as the implementation of the Onetrack system may alter the current procedure steps. The majority of the statements found in Appendix C are applied in the KBS. When the KA has been successfully tested, then the KBS is ready to be tested in the CLIPS environment.

9.4 CLIPS Environment Testing

It is necessary to test the functions of the KBS and its decisions against the desired answers. Also, actual patient cases are used to test the KBS response to actual data. Testing is completed in two stages.

The first stage is a continuous growing data list of potential input scenarios to the KBS. Each rule in the KBS should be tested at least once. Thus, special events are created to ensure that all rules are tested. Accuracy is determined by comparing the actual output to the desired output. If the KBS halts without producing an output, the fault is tracked and the code is modified. The KBS is then tested again with that same input to ensure that it works. If the KBS provides an incorrect output, the fault is also tracked and the code modified. This is repeated until the desired sets of outputs are obtained and all the scenarios are completed. It is not possible to show all the iterations of the first stage. However, the input in Appendix G.1 is representative of a generic sample data set. When the KBS is functioning appropriately, it is time to move to the next testing stage.

In the second stage, two actual patient case data sets are utilized to create simulated input scenarios. This input data is converted into the correct KBS syntax. Each input corresponds to a result which is examined to determine whether the desired result is achieved. The accuracy of the KBS is determined by the percentage of time that the KBS results agree with the desired results. Preferably this should be near 100 percent, but anywhere from 90 to 100 percent is acceptable. To improve the accuracy of the KBS, it might be necessary to add, remove or modify some of the knowledge. These changes depend on how the KBS performs during testing.

9.5 Test Cases

Two patient cases are chosen as sources to generate two simulated test cases. The data is gathered by listening to recordings of the operations and examining the paper documents involved with the procedure. This provides both the actual case results and user input. The test cases are given to the 3D modeling team who then create a probable path trajectory for the case based on the MRI scans of each individual patient. The DSP data is merely simulated data that is strategically selected to test certain aspects of the KBS. The KBS is executed with the simulated data input sets. The KBS is accurate if its results are the same as the desired results and the actual case results.

Patient number one (Patient 1) is an exceptional case, in that it has some conflicting information in the input data set. The desired solution was obtained by presenting this specific case to the experts during the Task 2 of the Knowledge Acquisition procedure. Patient number two (Patient 2) is a straight forward case without complications. The input data set, in the correct KBS syntax, for Patient 1 and Patient 2 is found in Appendix G.2. The KBS test results are found in Appendix G.3. The identification numbers for

each assert in the input file corresponds to the identification numbers for each result in the KBS output.

9.6 Test Results

To determine the accuracy of the KBS, the results of the KBS are compared to the results of the actual case studies and to the desired results based on the data input. The accuracy of the KBS is calculated by determining the percentage of correct results. Furthermore, the path plan found by the KBS is compared to the predicted path plan. The following tables are generated to capture the comparison between the KBS results, actual case data results (the case study answers) and actual input (input to the KBS). Table 9.1 and Table 9.3 contain the summary of the test results for Patient 1 and Patient 2 respectively. Table 9.2 and Table 9.4 contain the comparison between predicted path plan and the found path plan for Patient 1 and Patient 2 respectively.

Table 9.1 and Table 9.3 contain the following set of information. Each data entry has an identification number that corresponds to the identification number for assertions in the input file. The input file is found in Appendix G.2. The KBS results for the Layer and Cell type at a specific depth are shown, as well as the state. The state indicates whether the result for identifying the start of a layer is absolutely true (solid), not to clear (fuzz) or whether that layer has already been recorded (done). The KBS results are then compared to the actual case data result. An “x” indicates that there is a precise match. An “o” indicates that there is no match and thus the KBS answer does not compare to the case study. A dash mark means that there is no data at that specific depth. This occurs because the KBS receives information independent of the input from the actual case data. Lastly the KBS results are compared to the desired results based on the

Actual Input. The knowledge engineer looks at the input, decides what the result should be according to KA, and determines whether the KBS answer is feasible.

Table 9.1 Patient 1 Test Results Summary.

Results					Actual Data		Actual Input	
#	Depth	Layer	Cell	State	Layer	Cell	Layer	Cell
4	50.6	ST	Popcorn	solid	x	x	x	x
5	51.4	ST	Popcorn	done	x	o	x	x
6	52.85	ST	none	done	-	-	x	x
7	55.345	Le	none	fuzz	-	-	x	x
8	56	ST	none	done	x	x	x	x
9	56.84	GPe	none	fuzz	-	-	x	x
10	57.2	GPe	none	done	x	x	x	x
11	60.2	ST	Popcorn	done	x	x	x	x
12	60.225	GPe	none	done	-	-	x	x
13	60.3	ST	Popcorn	done	x	x	x	x
14	61.8	Le	Border	done	x	x	x	x
15	62	GPe	HFD-P	done	x	x	x	x
16	62.3	none	none	none	o	o	x	x
17	62.61	GPe	none	done	-	-	x	x
18	63	GPe	HFD-P	done	x	x	x	x
19	63.3	GPe	LFD-B	done	x	x	x	x
20	63.9	none	none	none	-	-	x	x
21	63.9	GPe	HFD-P	done	x	x	x	x
22	64.1	Li	none	fuzz	-	-	x	x
23	64.2	GPe	Border	done	x	x	x	x
24	64.46	GPi	none	fuzz	-	-	x	x
25	65.1	GPe	HFD-P	done	x	x	x	x
26	65.3	GPe	HFD-P	done	x	x	x	x
27	66	Li	Border	done	o	x	x	x
28	66.4	GPi	none	done	o	o	x	x
29	67.3	Li	Border	done	x	x	x	x
30	68	GPi	Tonic	done	x	x	x	x
31	68.28	La	none	fuzz	-	-	x	x
32	68.3	GPi	Tonic	done	x	x	x	x
33	68.5	GPi	Pausing	done	x	x	x	x
34	69.4	GPi	Pausing	done	x	x	x	x
35	69.6	GPi	Bursting	done	x	x	x	x
36	70.5	GPi	Border	done	x	x	x	x
37	70.9	GPi	Chugging	done	x	x	x	x
38	71	GPi	Tonic	done	x	x	x	x
39	71.7	La	Border	done	o	x	x	x
40	71.9	GPi	none	done	x	x	x	x
41	72.7	La	Border	done	x	x	x	x
42	74	La	none	done	x	x	x	x
43	75.5	OT	none	solid	x	x	x	x

Table 9.2. Comparison of Predicted and Found Path Plan for Patient 1.

Predicted Path Plan		**KBS Found Path Plan**		
Layer	Depth	Layer	Depth	State
ST	43	ST	50.6	Solid
Le	53.85	Le	55.345	Fuzz
GPe	56.84	GPe	56.84	Fuzz
Li	63.61	Li	64.1	Fuzz
GPi	64.43	GPi	64.46	Fuzz
La	68.21	La	68.28	Fuzz
OT	75.70	OT	75.5	Solid
IC	76.92	IC	-	-

Table 9.3. Patient 2 Test Results Summary.

Result					Actual Data		Actual Input	
#	Depth	Layer	Cell	State	Layer	Cell	Layer	Cell
4	56.9	ST	Popcorn	solid	x	o	x	x
5	57	ST	none	done	x	o	x	x
6	60.22	ST	none	done	-	-	x	x
7	63.17	Le	none	fuzz	-	-	x	x
8	65.12	GPe	none	fuzz	-	-	x	x
9	66	GPe	Border	done	x	x	x	x
10	67.3	GPe	Border	done	x	x	x	x
11	67.6	GPe	HFD-P	done	x	x	x	x
12	67.9	GPe	HFD-P	done	x	x	x	x
13	68.4	GPe	HFD-P	done	x	o	x	x
14	69	GPe	Pausing	done	x	x	x	x
15	69.4	GPe	HFD-P	done	x	x	x	x
16	69.81	GPe	none	done	-	-	x	x
17	70.5	none	none	none	x	x	x	x
18	70.87	Li	none	fuzz	-	-	x	x
19	71.56	GPi	none	fuzz	-	-	x	x
20	71.7	Li	Border	done	o	x	x	x
21	70.5	GPe	LFD-B	done	x	x	x	x
22	71.8	Li	Border	done	o	x	x	x
23	71.9	Li	Border	done	o	o	x	x
24	72.8	GPi	none	done	o	x	x	x
25	72.8	GPi	Pausing	done	x	o	x	x
26	73.1	GPi	none	done	x	x	x	x
27	73.65	GPi	none	done	-	-	x	x
28	74.3	GPi	Injury	done	x	o	x	x
29	75.1	La	none	solid	o	x	x	x
30	81	none	none	none	x	x	x	x
31	80.4	none	none	done	x	x	x	x

Table 9.4. Comparison of Predicted and Found Path Plan for Patient 2.

Predicted Path Plan		**KBS Found Path Plan**		
Layer	Depth	Layer	Depth	State
ST	50.35	ST	56.9	Solid
Le	61.22	Le	63.17	Fuzz
GPe	65.12	GPe	65.12	Fuzz
Li	70.18	Li	70.87	Fuzz
GPi	71.56	GPi	71.56	Fuzz
La	75.72	La	75.1	Solid
OT	-	OT	-	-
IC	-	IC	-	-

The accuracy percentage in Table 9.1 for Patient 1, when the KBS is compared to the actual case data for *layer* is 87%, for *cell* is 90% and for the *actual input* it is 100%. The accuracy percentage for Patient 2 when the KBS is compared to the actual case data for *layer* is 80%, for *cell* is 75% and for *actual input* it is 100%. This indicates that under all circumstances the KBS behaved as the knowledge engineer had expected, depending on the information sent to the KBS. The deviations of the KBS results from the actual case data exist because:

1) The actual input sent to the KBS was not the same as what the actual case data would indicate. This was done in some cases to test another aspect of the KBS.
2) The KBS uses the path plan to determine at what depth the KBS should wakeup next. If no other information but depth is available when an event happens, then the KBS assumes that the path plan is correct. This may result in the situation where the KBS identifies a layer to early. However, the KBS indicates this in the found path plan by showing that the state of that layer is fuzzy.
3) Some of the data inputs are simulated data from the DSP. These inputs result in unique answers, which can be different from the actual data sets.

4) Lastly, if there is a significant shift of more than 2mm in the desired path trajectory to the actual path trajectory it will also cause the KBS results to deviate from the actual data sets. Therefore it is important to obtain an accurate path trajectory from the 3D model.

When these possibilities are all taken into consideration, the accuracy percentage of the KBS with respect to the actual case data increases to approximately 95%. This is determined by calculating the number of expected results versus the actual KBS results.

Assuming that the three dimensional model and data from DSP is accurate, the KBS should be able to complete the mapping in one track. This is true if the error between the predicted path plan and actual path plan is small. Table 9.2. and 9.4. show the comparison between the predicted path plan and the actual path plan. Table 9.5. indicates the differences between the predicted path plan and actual path plan for each patient.

Table 9.5. Difference between predicted and actual path plan.

Layer	Patient 1	Patient 2	Average
ST	6.55	7.6	7.075
Le	1.95	1.495	1.7225
GPe	0	0	0
Li	0.69	0.49	0.59
GPi	0	0.03	0.015
La	0.62	0.07	0.345

The reason the ST has such a large error is that the user may start the track anywhere in the Striatum layer and not necessarily at its beginning. The possible reason for a large deviation of Le is because the lamina is difficult to distinguish. However, for the differences between path plans for the other layers are relatively small, less than 1mm. Therefore it is possible to assume that the KBS is accurate. If the accuracy of the data

sources are high the accuracy of the KBS will be high and therefore the user will only need to execute one track to get an understanding of the probe location relative to the Basal Ganglia. Note however, that this conclusion is based on only two test cases. More test cases of the entire Onetrack system should be completed in order to validate this statement.

The actual path trajectories for both patient cases are acceptable. This information is made available to the user graphically. On the screen the user will see a plot of both the predicted path trajectory, the KBS actual path trajectory, the data points created by the KBS and the data points created by the user. Remember that the KBS is merely a guidance system for the user. It is ultimately still the user's responsibility to determine the orientation of the track in the three dimensional model space and to decide where to place the next track.

All in all, the KBS is found to be acceptable and ready to be implemented into the Onetrack system.

9.7 Other Tests

The method of testing the KBS in the Onetrack system environment, real time operation testing and expert testing is described next. These tests can only be performed once the Onetrack system is completed and ready to be tested in its entirety. The KBS is stable and should be able to function well within the Onetrack system as long as the communication protocol is followed.

The KBS interaction with the Onetrack system should to be tested first. In other words the data exchanges between the two systems need to be tested. This is done in two stages. The first stage involves choosing random case scenarios to test out the interface. If the data exchange is in the correct format then the KBS output should be the same as when it is executed in the CLIPS environment. Debugging is difficult while running the KBS integrated with the Onetrack system, because the facts that are being generated are not available to view. If the KBS does not send information back to the main program of the Onetrack system then there is probably a problem in the data exchange between the programs. It can not be the KBS process knowledge because that has already been tested and proven to be reliable during the CLIPS environment test. The data exchange problem should be identified and resolved before proceeding to the next stage. Then the same scenario as what is executed to determine if it works. This process is repeated until all the random case scenarios are completed.

Once the interface communication is stable, it is necessary to move to the next stage of testing. This involves generating the two simulated patient cases by the other components of the Onetrack system. The Onetrack system feeds information to the KBS during a real time simulation. Once again the event results and actual path plan are compared to the actual and desired results. The accuracy of the KBS then depends on the information sent and the results reliability.

Ultimately the entire Onetrack system will be taken into the operating room and a real time operation test performed. The success of the Onetrack system will be measured by how well its results compare to the current system used by the experts. The accuracy of the KBS depends on how well it makes decisions and how these decisions agree or

differ with the user. This test will determine how well the KBS mapped the actual track and how well this compares to the maps generated by the user. If these are identical, or if the KBS is better, then the results will be acceptable. However, if there is a deviation of a few millimeters, then improvements to the KBS will be required.

The final step in testing the KBS is to allow an expert (neurologist) to test it. For ease of use the expert should be given the entire Onetrack program (which should incorporate the KBS.) Based on the expert's opinion of the entire system, adjustments should be made as necessary.

9.8 Summary

The KBS is able to adequately accomplish the tasks that it was created for. Both tests yielded desirable results. The KBS reflects the knowledge gathered during KA, in that it imitates the process of the operation and contains facts important to the reasoning process. The CLIPS environment testing is useful for two main reasons. First, the tests were able to identify some areas of weakness in the KBS, which were taken care of immediately. Secondly, it illustrated that the KBS works correctly given the design constraints and data inputs. The KBS is ready to be tested in the fully integrated Onetrack system and then tested in the operating room.

CHAPTER 10

CONCLUSION

10.1 Introduction

The KBS plays an important role in the Onetrack system, in that it provides guidance to the surgical team to execute the procedure. A summary of this thesis is presented in this chapter followed by a discussion on probable applications in Civil Engineering. Possible improvements that could be made to the KBS are discussed in the future work section.

10.2 KBS Summary

The hypothesis in Chapter 1 stated that

> *The central premise of this research is that a knowledge based system can be developed to assist in guiding the surgical team with a Pallidotomy or DBS procedure.*

This statement is supported throughout this thesis. The evaluation of the KBS (Chapter 9) indicates that the KBS is capable of providing the neurologist with an accurate identification of the layer and cell type that the probe tip is in at a given depth and point in time. The KBS is capable of making this information known to the neurologist through the Onetrack system in a graphical format (Chapter 8). The neurologist is then capable of determining where the actual path plan is with respect to the predicted path plan and brain anatomy, which leads to a higher accuracy of future track execution. The KBS is capable of keeping track of the decisions made throughout the procedure; therefore it

improves the reasoning process of the neurologist because most relevant data can be found in the program. These advances aid the neurologist in such a way that the time the procedure takes is reduced and therefore the risk of infection is reduced. Another risk that the KBS reduces is the chance for patient blindness. The KBS keeps track of the location of the probe tip and will notify the neurologist when the probe tip is close to the Optic Track (Chapter 7.) Without the KBS, the Onetrack system would merely be an analytical tool that responds to data. The KBS is therefore a vital component of the Onetrack system, because it causes the Onetrack system to be a guidance tool (Chapter 7).

Parkinson's Disease (PD) is caused by a deficiency of dopamine in the Globus Pallidus Interior (GPi). L-dopa medication is the standard treatment for Parkinson's symptoms. However, L-dopa does not work long term in all patients. In these situations a surgical treatment can be performed to alleviate symptoms for some patients. These treatments are a Pallidotomy (a lesion in the GPi) or Deep Brain Stimulus implantation of an electrical device in the GPi. The Onetrack system is created to aid the neurologist (user) in performing these procedures. The Onetrack system components include a main program that is the direct interface with the user, a three dimensional model of the brain, a digital signal processor and a knowledge based system (KBS). The components of the Onetrack system are able to determine the position of the probe relative to the patient's brain anatomy (Chapter 3). The role of the KBS is to interpret the data made available by the other components, and determine the anatomical layer and cell type classification of the probe at a specific depth and a given point of time during a track execution (Chapter 7).

A KBS is not an algorithm (Chapter4), but instead a clear linguistic representation of the knowledge of the expert (Chapter 5). The knowledge that exists in the KBS is taken from experts in the field that have performed multiple Pallidotomy and DBS procedures (Chapter 6). Knowledge is also gathered through literature review and examination of patient cases. This knowledge is then encoded to create a KBS. First the process knowledge, the order of activities involved with the experts reasoning, is identified and encoded. Then all the other pieces of knowledge are added to create the domain knowledge, therefore adding functionality to the KBS (Chapter 7).

The KBS and the main program of the Onetrack system are written in two different environments. A communication protocol is created to allow interactions between these two programs (Chapter 8). The integration of the KBS with the Onetrack system is completed once the language is set and data can successfully be interpreted by both environments.

Two tests are done to determine the appropriateness of the KBS. First, the knowledge encoded in the KBS is examined and compared to the knowledge gathered from the experts. The process knowledge reflects the process that is indicated by the experts. However, there are some differences. For example, the addition of a path trajectory that is determined by a 3D model and a probability array of cell types produced by the DSP. These sources of information are currently not available to the user. So the knowledge engineer determined how this information should be interpreted in the Onetrack system. The second test is to run the KBS in the CLIPS environment with two simulated test cases. The accuracy of the KBS is dependant on how closely the KBS decisions reflect

the actual decisions made by the expert in the cases. The accuracy of the KBS is found to be appropriate.

10.3 Application in Civil Engineering

The design cycle of the KBS is similar to the design cycle involved with any Civil Engineering problem. First the problem is analyzed (Knowledge Acquisition). Then based on analysis a first rough design is generated (First KBS created). Analysis is completed to see whether the design is appropriate for the analytical results (KBS testing). If the design is inappropriate, the problem is redesigned (Second KBS created). This iterative process continues until an ultimate solution is reached (Ultimate KBS). The design is then implemented (the KBS is integrated with the Onetrack system). There are multiple applications of the KBS in Civil Engineering. Only two possible applications are discussed here.

The process knowledge of the KBS can be potentially used in the guidance of oil drilling. The scale of the problem is obviously different. In the brain the probing is on the scale of a few millimeters, but with oil drilling it could be several kilometers. However, in oil drilling the user also deals with determining what layer the drill bit is in at a given depth and point in time. Also, the user looks for the most optimal position to place the drill, utilizing seismology as an input to the optimal decision. “Finally, and most commonly, they use seismology, creating shock waves that pass through hidden rock layers and interpreting the waves that are reflected back to the surface” [31.] It is necessary to identify the different soil layers, in order to determine the optimum position of the drill. The oil drilling domain knowledge is different, but the problem structure is similar to that of the presented medical procedures.

KBS's are also making their way into earthquake engineering. The Modified Mercalli (MM) intensity scale is currently used to determine the intensity of an earthquake based on the level of destruction. Level of destruction is defined by numerous parameters, such as level of furniture destruction, what the person felt when the earthquake hit, and other various heuristical data sets. "The intensity scale consists of a series of certain key responses such as people awakening, movement of furniture, damage to chimneys, and finally – total destruction" [32.] A general KBS model can be used to help identify the level of intensity of an earthquake, by having people fill out a questionnaire generated by the KBS. The primary component of the Onetrack KBS that could be used is the communication protocol between the main user interface program and the KBS driving the questionnaire.

10.4 Future Work

There are three areas in which the KBS can be improved. These are uncertainty representation, historical reasoning and motor-sensory mapping.

There are several sources of uncertainty within the Onetrack system. For example, equipment calibration, data measurement errors, ignal pattern interpretation, MRI inaccuracies and three dimensional patient specific brain models. One example of equipment uncertainty parameters is the shape of the probe tip that causes the track to be slightly off from the predicted path trajectory, even though all the other settings are accurate. The KBS is not currently capable of dealing with uncertainties within the data sent to it. This is definitely an area that can be further investigated, potentially leading to a major improvement.

Currently the KBS only compares the current data set to the previous data set. It would be an improvement if the KBS could compare the current data set to that found in a previous track or a previous operation performed on the patient. This would be a broadening of the domain knowledge of the KBS. The process knowledge will remain the same, with the addition of referencing previous tracks.

Last, one important source of information that the Onetrack system does not incorporate in the reasoning process is the motor-sensory exploration performed by the user. The Onetrack system currently captures this information, but it does not influence the guidance given by the Onetrack system. The Knowledge Acquisition proves that motor-sensory exploration plays an important role in determining the anterior, posterior, medial or lateral position of a single data point along a given track. A method of actually capturing patient movement will have to be devised, such as attaching a monitor to a patient's arm that could measure the tremor of the arm. If the frequency of the tremor is the same as the frequency of discharge in the neuronal signal, then certain conclusions can be drawn as to the current probe layer. Currently all of this information is not captured or utilized.

10.5 Summary

The purpose of the KBS is to provide the user with an accurate decision on the layer and cell type of the probe at a specific depth in the brain at a given point in time. The KBS is developed by gathering knowledge from experts and encoding these knowledge statements. The KBS is tested to ensure that it corresponds to the knowledge gathered from the experts. The KBS is only one component of the Onetrack system, and as such has to function with other components of the Onetrack system. Possible applications of

the KBS in the Civil Engineering field include transformation of the domain knowledge to aid in the decision process involved with drilling for oil. The three main areas in which the KBS may be improved are representing uncertainties within the Onetrack system, reasoning with previous knowledge and incorporating motor-sensory exploration.

APPENDICES

APPENDIX A

QUESTIONNAIRE USED IN INTERVIEWS.

Knowledge Acquisition for a Pallidotomy

Performed by:

Dr. Nelson Baker

Ms. Linda Harley

Introduction

The purposes for today's meeting are to validate existing knowledge, determine strengths of knowledge (what knowledge to believe more strongly than others), and find new knowledge. Standard techniques for knowledge acquisition will be used. For this reason we will not be showing you the current technological tools. At a later time we will be presenting the actual tool that is being created.

Three activities will be performed.

1) Practice session.
2) Walk through a track using actual patient records. More details will be provided.
3) Follow up questioning.

During the entire process we would like you to talk aloud. We will remain silent throughout the process except by reminding you to keep talking aloud. You will be prompted by written instructions.

Do you have any questions?

TASK 1

Practice Session

Please read the following instructions aloud. Please being and remember to talk aloud.

Introduction

"In this knowledge acquisition we are interested in what you say to yourself as you perform a task that we give you. In order to do this, we will ask you to talk aloud as you work on the problems. What I mean by talk aloud is that I want you to say out loud everything that you say to yourself silently. Just act as if you are alon in the room speaking to yourself" or as if you are explaining things to a group of residents. "If you are silent for any length of time I will remind you to keep talking aloud. Do you understand what I want you to do?" [1, Appendix].

Before we turn to the Pallidotomy knowledge acquisition, we will start with a practice problem.

Let's Begin

I want you to talk aloud while you do this problem.

Warm up: I would like you to solve the anagram below. It is your task to find several words that consists of all the presented letters. For example, if the scrambled letter are KORO, you may see that these letters spell the word ROOK. Any questions? Please "talk aloud' while you solve the following anagram.

SCIOEN

TASK 2

Walk through of track.

Please read the following instructions aloud. Please begin and remember to speak aloud. Please explain your reasoning and answer each of these questions.

Question 1:

During a Pallidotomy operation what is your overall strategy for a DBS or Lesion?

Question 2a:

What setup procedures, initializations, and precautions are you concerned with prior to the first track?

Question 2b:

Is there anything you check/validate in the operation room prior to track execution?

Question 3:

This question involves looking through patient records of a track procedure. (Patient records are provided at the end of this task.)

- Given the following 3 pages of information from an actual patient's Pallidotomy, please describe to us the first track execution.
- Limit the discussion to the time from when the probe enters the brain until the point that the probe is retracted.
- Include explanations and reasoning for
    - Patient interaction,
    - Depth observation,

- Cell type classification,
- Layer determination, and
- Precautions taken throughout the procedure.

Please talk aloud and explain your thoughts as you would to a group of new residents.

Question 4:

Given the next two pages does this reinforce or change your belief or findings from Question 3?

Question 5:

What activities are involved in deciding where the next track should be placed?

Question 6:

What would be your recommendation for where the next track should be placed with the information provided in Question 3?

Question 7:

Anything else you would like to add regarding track execution/procedure?

TASK 3

Follow up questions.

Please read the following instruction aloud. Please begin and remember to talk aloud. In this task we would like you to answer the following questions pertaining to a Pallidotomy procedure. Please explain your reasoning and answer each of these questions.

Question 1a:

How do you determine when you have crossed into a new anatomical layer?

Question 1b:

What factors do you include when deciding which anatomical layer the tip of the probe is located in?

Question 2a:

What kinds of uncertainties exist in determining the probe location in a given track? (z-depth.) You may list the answers as you talk aloud.

Question 2b:

To what extent (small or large degree) do these uncertainties (listed above) exist in the determination of the probe location in a given track?

Question 2c:

What kinds of uncertainties exist in determining the track placement? (3 dimensional mapping.) You may list the answers as you talk aloud.

Question 2d:

To what extent (small or large degree) do these uncertainties (listed above) exist in the determination of the track placement?

Question 3:

How do you determine if and when patient interaction is needed?

Question 4a:

What types of patient interaction exist and when do you use each? You may list the answers as you talk aloud.

Question 4b:

Are there uncertainties associated with these factors? If so, please describe the uncertainty and its extent (small/large).

Question 5a:

When listening to the audio signal during a track, how do you determine what cell type you hear?

Question 5b:

Can you associate cell type with layer? If so, what are the associations?

Question 5c:

What cell types are difficult to discern/identify when listening to the audio signal?

Question 6a:

What tools are used in determining where to place the next track and what information from the current track do you use in making this decision?

Question 6b:

When performing subsequent tracks what information do you use from previous tracks to understand the current track?

Question 7a:

Under what conditions do you pause the procedure and what actions do you perform?

Question 7b:

Under what conditions do you abort the procedure and what actions do you perform?

Question 7c:

Under what conditions do you restart the procedure and what actions do you perform?

Question 8:

In the ideal world what kind of technological assistance would you like to see to improve the process?

Question 9:

How do you anticipate using the new technology in tracking?

Question 10:

Now that you have completed both tasks, is there anything else that you would like to include that should be captured by our system?

Discussion

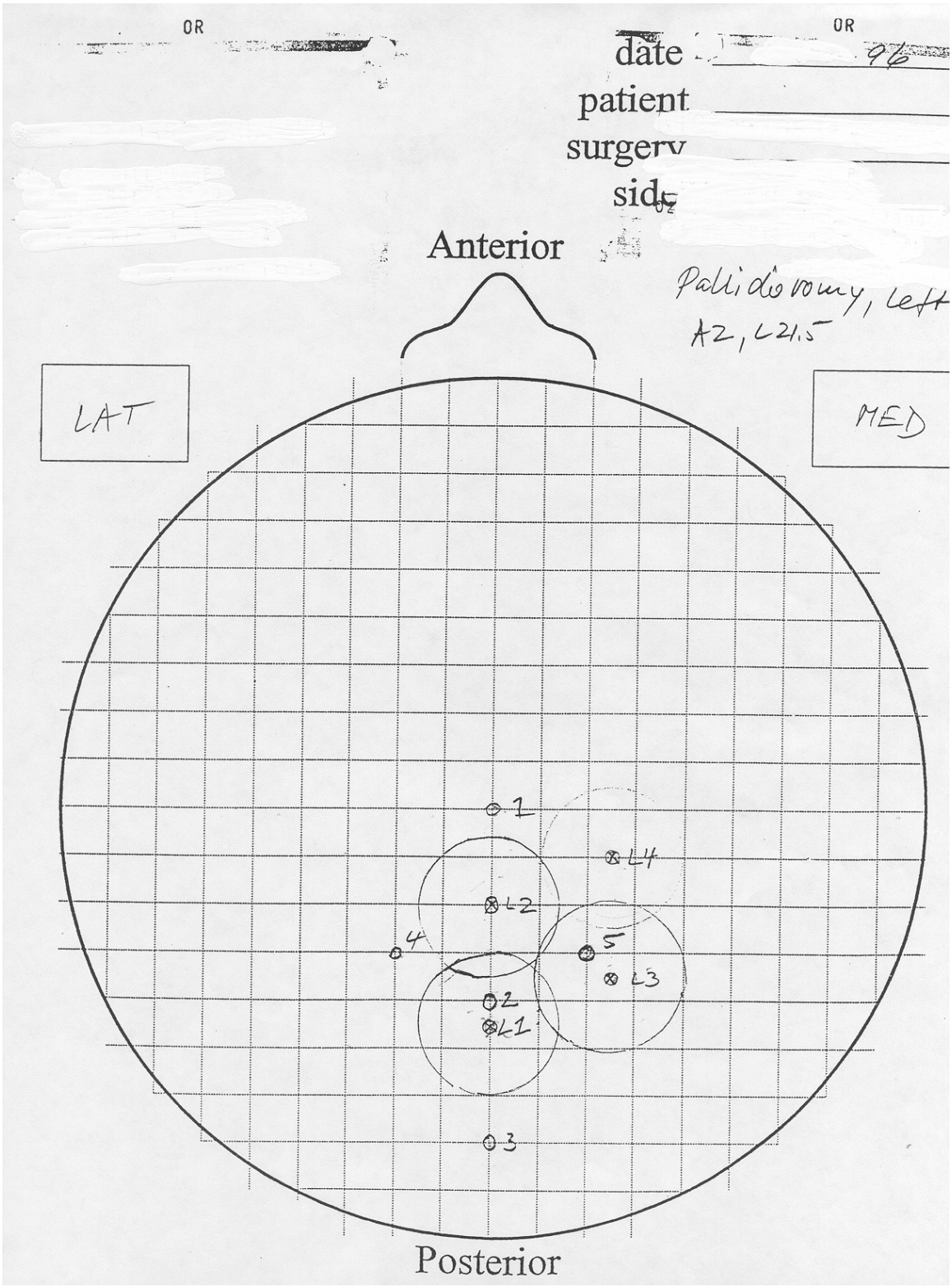


Figure A.1. The first figure used in Task 2 Question 3.

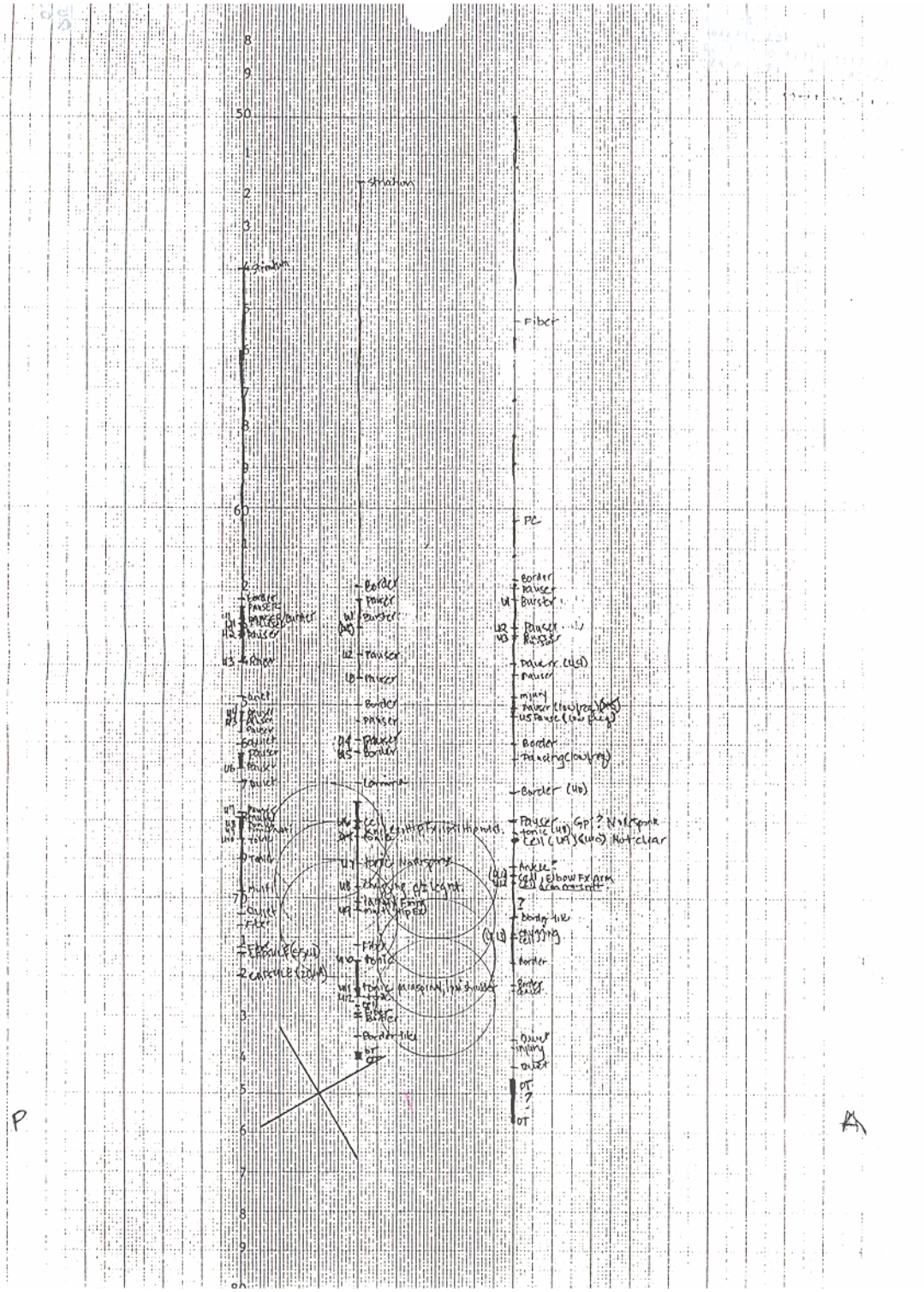
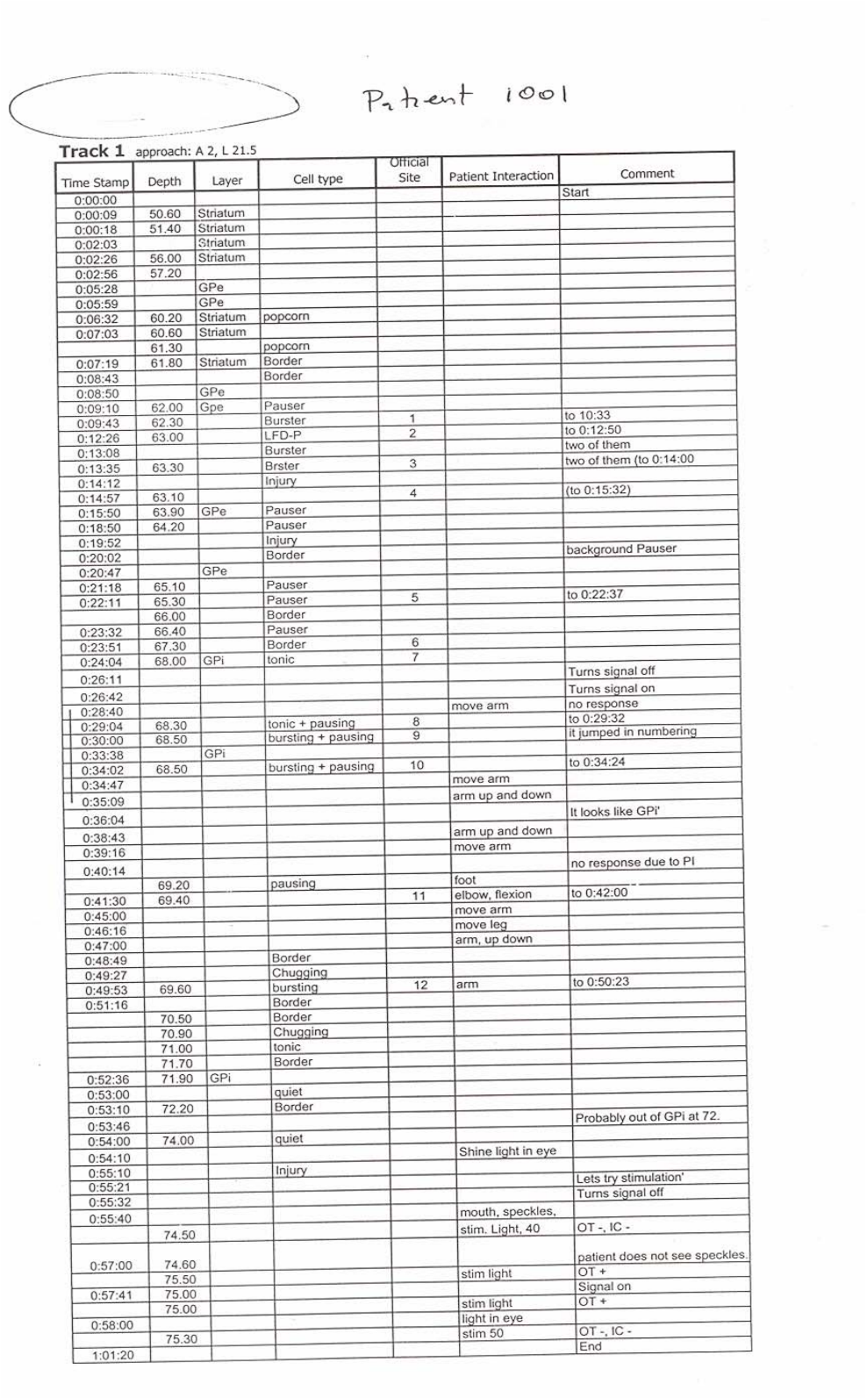


Figure A.2. The second figure used in Task 2 Question 3.



Patient 1001

Track 1 approach: A 2, L 21.5

Time Stamp	Depth	Layer	Cell type	Official Site	Patient Interaction	Comment
0:00:00						Start
0:00:09	50.60	Striatum				
0:00:18	51.40	Striatum				
0:02:03		Striatum				
0:02:26	56.00	Striatum				
0:02:56	57.20					
0:05:28		GPe				
0:05:59		GPe				
0:06:32	60.20	Striatum	popcorn			
0:07:03	60.60	Striatum				
	61.30		popcorn			
0:07:19	61.80	Striatum	Border			
0:08:43			Border			
0:08:50		GPe				
0:09:10	62.00	Gpe	Pauser			
0:09:43	62.30		Burster	1		to 10:33
0:12:26	63.00		LFD-P	2		to 0:12:50
0:13:08			Burster			two of them
0:13:35	63.30		Brster	3		two of them (to 0:14:00
0:14:12			Injury			
0:14:57	63.10			4		(to 0:15:32)
0:15:50	63.90	GPe	Pauser			
0:18:50	64.20		Pauser			
0:19:52			Injury			
0:20:02			Border			background Pauser
0:20:47		GPe				
0:21:18	65.10		Pauser			
0:22:11	65.30		Pauser	5		to 0:22:37
	66.00		Border			
0:23:32	66.40		Pauser			
0:23:51	67.30		Border	6		
0:24:04	68.00	GPi	tonic	7		
0:26:11						Turns signal off
0:26:42						Turns signal on
0:28:40					move arm	no response
0:29:04	68.30		tonic + pausing	8		to 0:29:32
0:30:00	68.50		bursting + pausing	9		it jumped in numbering
0:33:38		GPi				
0:34:02	68.50		bursting + pausing	10		to 0:34:24
0:34:47					move arm	
0:35:09					arm up and down	
0:36:04						It looks like GPi'
0:38:43					arm up and down	
0:39:16					move arm	
0:40:14						no response due to PI
	69.20		pausing		foot	
0:41:30	69.40			11	elbow, flexion	to 0:42:00
0:45:00					move arm	
0:46:16					move leg	
0:47:00					arm, up down	
0:48:49			Border			
0:49:27			Chugging			
0:49:53	69.60		bursting	12	arm	to 0:50:23
0:51:16			Border			
	70.50		Border			
	70.90		Chugging			
	71.00		tonic			
	71.70		Border			
0:52:36	71.90	GPi				
0:53:00			quiet			
0:53:10	72.20		Border			
0:53:46						Probably out of GPi at 72.
0:54:00	74.00		quiet			
0:54:10					Shine light in eye	
0:55:10			Injury			
0:55:21						Lets try stimulation'
0:55:32						Turns signal off
0:55:40					mouth, speckles,	
	74.50				stim. Light, 40	OT -, IC -
0:57:00	74.60					patient does not see speckles.
	75.50				stim light	OT +
0:57:41	75.00					Signal on
	75.00				stim light	OT +
0:58:00					light in eye	
	75.30				stim 50	OT -, IC -
1:01:20						End

Figure A.3. The third figure used in Task 2 Question 3.

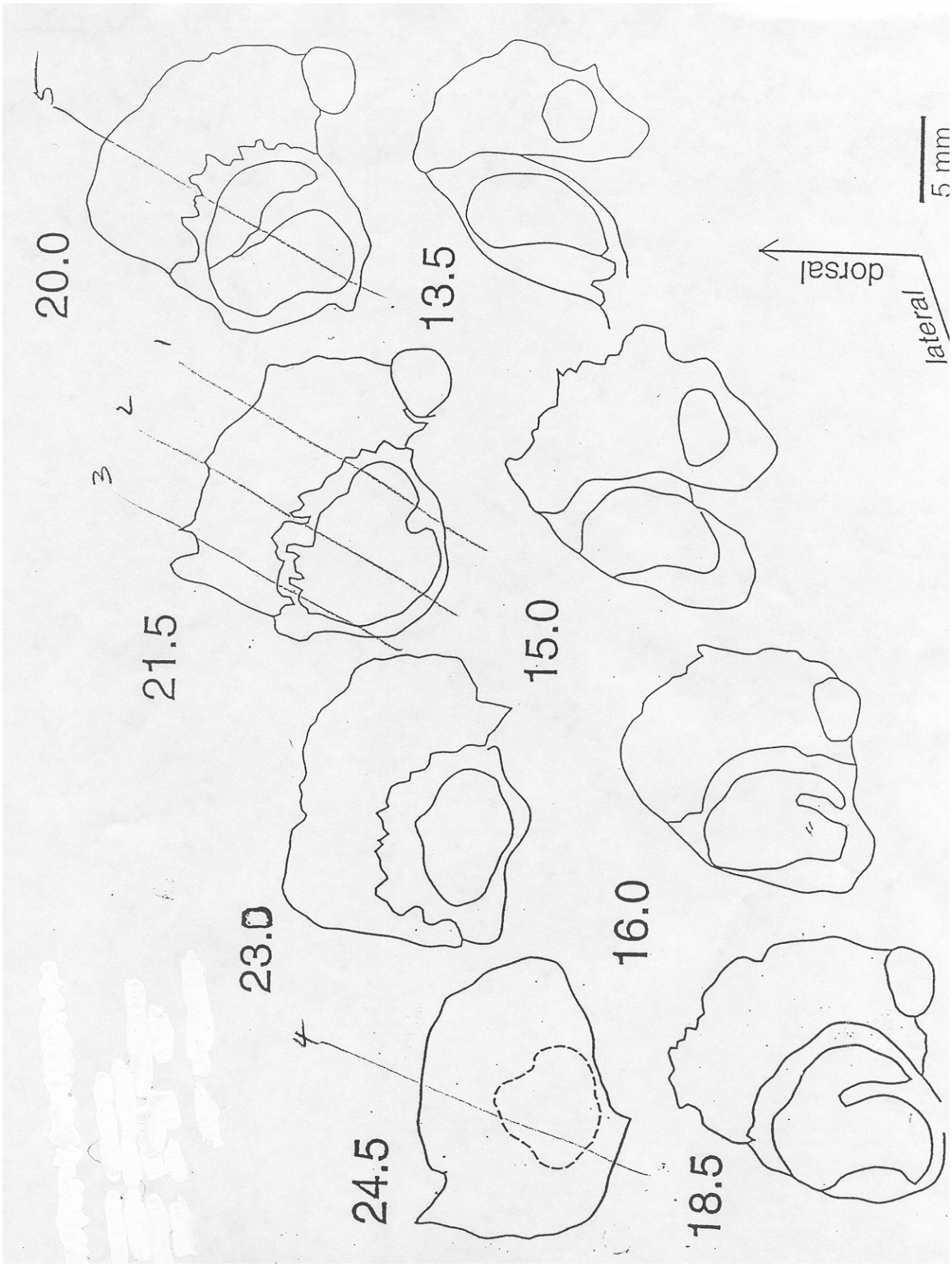


Figure A.4. The first figure used in Task 2 Question 4.

Name _____ Date 96 Side L

Lesion Tract # 1 Approach P2.5, L2.5

Stimulation

#	Depth (mm)	Curr (mA)	Frq (Hz)	Time	Response			
					Vol Move & Tone	Invol Move	Capsular	Visual
1	69.0		300			1st finger adduct.	1mA	
2	70.0		"			index finger	.7mA	
3	71.0		"			"	.6mA	
4	72.0		"			"	.5	
5	73.0		"			"	.4mA	
6	72.5		"				.4mA	
7	72.0		"				.5mA	>1.4

Lesioning

L#	Depth (mm)	Temp (°C)	Response			
			Vol Move	Tone	Invol Move	Others
1	72.0	60			arm dysthesia (slight)	
2	72.0	70				
3	71.0	75				
4	70.0					

Figure A.5. The second figure used in Task 2 Question 4.

APPENDIX B

TRANSCRIPTS OF INTERVIEWS

B.1 Interview with Expert 1

Knowledge Engineer 1:

We are here today to for plain knowledge acquisition. You can just read through the instructions. Our task is to validate any of the info we have used and to gather new information if necessary

Knowledge Engineer 2:

These are the same forms that we will pass to Expert 2. They will be identical.

Expert 1:

Ok I am just going to go ahead and answer your questions then. Practice session. So you want me to read that.

Knowledge Engineer 2:

Everything we want you to do aloud and probably a little louder talking voice since the camera is over here.

TASK 1

Expert 1:

Introduction in this KA we are interested in what you say to yourself as you perform a task that we give you. In order to do this we will ask you to talk aloud as you work on the problems. What I mean by talk aloud is that I want you to say out loud everything that you say to yourself silently. Just act as if you are alone in a room speaking to yourself or as if you are explaining things to a group to a group of residence. If you are silent for any length of time I will remind you to keep talking aloud. Do you understand what I want you to do. Yes I do. Yes I do. Ok. Now I should be good at that because I usually talk to myself when I do things. I'll just be myself then.

Before we turn to Pallidotomy ka we will start with a practice problem. Lets begin I want you talk as you while you do this problem. Warm up. I would like you to solve an anagram below. Ok. As if you where task as if your task to find several.. oh sorry.. it is your task to find several words that consist of all the present letters. For Examples if the scrambled letter is koro you may see that these letters spells the word rook any questions please talk aloud while you solve the following anagram. Ok. Umm.

Knowledge Engineer 2:

So talk aloud…what are you thinking.

Expert 1:

I'm trying. Wow what is this here. That's difficult. Ok. It is not science because it has an o in there. Ok.

Knowledge Engineer 2:

What are you trying to do?

Expert 1:

Well what I am trying to do. I am trying to you know. shuffle these letters around. You see. To Find the first letter that that would start a word that I know. Neo.

Knowledge Engineer 2:

Well say which letter you are using as the first one. And how you are going about the problem solving.

Expert 1:

Ok. N. so I start n, like no new neo I am not good at those things. Neos. Neosie no that is definitely not a word. Um. C. um That's.

Knowledge Engineer 2:

What are you trying now?

Expert 1:

I am trying to shuffle these letters around. I mean.

Knowledge Engineer 2:

Ok but keep talking about which letter you are trying to use first. That's why this is a practice we want you to say what you are thinking.

Expert 1:

Ok. Is this an easy word or not. or is this some kind of unusual word.

Knowledge Engineer 1:

It could be several words.

Expert 1:

It could be several words. Oh really, that not an

Knowledge Engineer 2:

Well you could make several words out of it using any of the letters.

Expert 1:

Lets see. I am using o. I am going through and just start using each letter and then scramble the others around like a brute force method. So cenic no Ceuno, could you have given me something more easy to warm up on, this is a little bit hard.

Knowledge Engineer 1:

Well for example you have an s i and n then you can spell the word sin. Or you have n o s e so you can spell the word nose.

Expert 1:

Um I see. I don't have to use all of them.

Knowledge Engineer 2:

Not necessarily. It is how many words can you find in it.

Expert 1:

Ohh I see. Uuuhh. So that kind of so that was my misunderstanding. I tried to make it like cause you took koro rook and you used all the letters there you know and that is why I assumed that I had to use all the letters for the word. That's why. ok.

You could have neo for instance, for neo. You could have Uhh don't have soo. Since. Since then. Ok Well alright I think that is plenty. Do I have to find all of them.

Knowledge Engineer 2:

No we are not looking for all but we want you to articulate your thought process. As to which letter you are using first. How you are trying to combine it.

Expert 1:

Well I used common letters, words start with certain common letters more commonly than others. So I used those like n is a very common one. S is a common one. And since I am bilingual I'm thinking a little of different ways about things about how it works than other people. Since English is not my mother language I I I am not that good in these things anyways. ok. All right.

Task 2

Expert 1:

Walk through the track ok. Uh. Task

Question 1. During a Pallidotomy operation what is your overall strategy for a DBS or lesion. Ok. Umm. First of all. When we start the track um what we do first of all uh. After the targeting is done, after everything is uhm essentially setup to work correctly and properly. I am assuming that you know testing the electrode and all that stuff is already done because you are not getting involved in that. Uhh. So first thing you do is look for the first target that you expect to encounter which would be uh the striatum in our case. We start high enough so that we are most likely find striatum. Um that is for us an important thing because Striatum, the extend of striatum that we find will tell us very much how lateral or medial we are in respect you know to our target our our preferred target location, um. The more medial we are from the anatomy we expect to encounter caudate on the top and a, that is usually the highest structure in the striatum and uh if we encounter, we can encounter a stretch of 5 6 maybe 8 millimeters of uh uh striatum very high up lets say relatively high up then we know we know we are medial and if we then further go down it will drop out we will be medial and only a few fingers of striatum will occur. But if we are more lateral we will get a deeper and get more of it and then the second, uh the second information we gain from striatum is that the more anterior we are the longer the run will be through striatum. In any case. And the more posterior we are the more uh the less we will get of a run. So so the striatum will give us the first indication of our medial lateral anterior posterior extend. If we don't get any striatum at all that that ill will tell us two things. Either our electrode is not performing correctly that we you know that it could it could always be that we don't get any cells because there is

something wrong. Even with the electrode tested out ok it could still be a problem with the uh with the recording later on. Could get damage by some reason for some reason. Um. Then we go further down and try to encounter all the all the other structures. Now the overall strategy is to uh to map the basal ganglia as thoroughly as possible with the least amount of tracks. And then secondly to find physiological or physiology information that is due to the sensory motor exploration that give us an indication where in the striatum. I mean not striatum. In Basal ganglia or in the GPi we are. So that we know which parts we actually are wanting to lesion or put the DBS lead in. because if a patient for instance has arm um, problems with the arm or with the leg or legs then we would preferably be in the arm or the leg area and then with the sensory motor exploration we can actually explore that and find the area that is actually responding to the arm or to the leg. And then specifically focus on that um. So this is our strategy. And then the third, thirdly is our strategy is to do the least amount of damage to vital parts that are not part of the of our target area like the optic track and the internal capsule to avoid those. Uh not to cause any lesions there and any deficits. So that is the overall strategy for. And uh and that is for surgery the strategy is of course to cause the most precise placement and then cause the symptoms to go away. To alleviate the symptoms.

So ok. Setup procedures What setup procedures, initializations and precautions are you concerned with prior to the first track? Um well the setup procedures of course is to target, put the frame on straight, target the the structure very precisely so that we are actually where we want want to be which is within the limitation of course the MRI the resolution of it and the prediction from the anatomy where actually is the right location that's why we do anatomy anatomical mapping and sensory motor mapping. Because a

sensory motor gives us another piece of information that the anatomy the MRI cannot tell us because it only tells us structures but it does not tell us uh whether these structures are involved with the limb whether we are really in the sensory motor area of the of the structure not in the associative or … part. That's why the mapping is so so important that the micro electrode recording is the only tool that actually can give you this information. Stimulation or just targeting does not give you any of that information. Um. Precautions is of course to um try to limit the number of track. Try to do the tracks as expediently as as fast as possible so to not cause prolonged exposure of the brain and cause bleeding cause the longer your have the craniotomy open the longer you have the electrode you have the electrode in the brain the more likely you have seizure can occur occurring hemorrhaging occurring all these things are very uh threatening to the patient so we try to reduce that risk. Uhh so and then uh just technical things to make sure that the um electrode does not resonate to much we get don't get to much what we call micro phonics which is most interference we put glue type glue biocompatible in the opening that is something we also do. And just to make sure that the equipment is working right. I mean that's another thing. We test it out before running a track we test out the impedance of the electrode before we even go down to condition the electrode so that the electrode is in prestean as far as we know at that point of time is in good working or excellent working condition. And the impedance of the electrode reading will tell us some about that. Ok that was question 2 a . moving on to question 2b. is there anything we check, validate, in the operating room prior to the track execution? Now I have already answered some of those questions. It is the the equipment of course gets setup by the technician before hand. Everything is there is a test procedure for that. which is attached to the rack which they go through and test out all the step by step conformation that all the components are working. And of course we

check out the electrode as I said when we start the track. And uh we do that high up in the brain when you stimulate the electrode you condition it you blow out a little bit of glass and we do not want and it also could cause a small lesion there we don't want to lesion close to the areas that we want to record. So we do that high up. Um.

Ok question 3. this question involves looking through patient records of a track procedure. You have given the following three pages of information from an actual patients Pallidotomy. Please describe us to us the first track execution. Limit this discussion from the time that the probe enters the brain until the point that the probe is retracted. Include explanations and reasoning for patient interaction depth observations, cell type classification, layer determination and precautions taken throughout the procedure. Please talk aloud explaining your thoughts as you would do as you would do to a new group of residence. Ok. Oh that is small. I need some glasses for that. Ok. So uh timestamp zero. At about 9 seconds we get the first reading at depth 50.6 of striatum. And uh another at 51.40 so they are close together, um that is 9 seconds later. Um. So this seems to be like a continues striatum there. So .8mm down so we consider that continuous. And that is important because if you have interruptions and quiet some areas between longer stretches between one cell encountered and the next one in striatum we call that non continuous. It means it is like a piece, a patch of striatum and then there is only white matter around it, just the capsule. Then you go down and you find another cell, if there is not more, 3 4 continuous in a row you either the electrode is not picking up, there is always a possibility that the electrode is not that sensitive. Some of them are not that sensitive to pick them up. And then uh you want to know when you have continuous striatum or whether you have patches of them, which is important cause in-between the caudate and the Putamen there is this fingered area of

interdigitating, we call it. That uh white and grey matter which is part is actually the striatum and the other part is internal capsule. And when you get into that area it becomes less continuous and that is an indication also of where you are in term s of laterality. Um. And anterior posteriors extend. At so then between 51.40 to 56 there is uh a label there's no this label is striatum, but there is no. um. *(interruption by phone)* The uh there is no depth label there, so I do not know in this case there was a reading or not in-between 51.4 51.4 and 56. that would be important. I would assume that there is a striatum there. That could be. Is this an original record, or is this a um. Your not suppose to answer? Ok. So my feeling is that there is a non continuous striatum there. Could either that the person making the notes did not bother to put everything there or didn't bother to write in-continuous for that. So we don't know at this point whether it is continuous or not so it has to be left open. The last reading I assume striatum is 57.2 and then there is with the other depth. That is 2min and 56 seconds later. Then without a depth reading at 5 min 28 there is GPe and 5:59 there is another GPe. So I think that that is then surprisingly at 60.2 there is again striatum that says popcorn. So that GPe must be a wrong reading there. Just because the GPe can not be inter mixed with Striatum. So that I just essentially disregard the GPe. So at 62.2 there is another Striatum popcorn. 60.6 there is another striatum. And then 61.3 there is another um popcorn. Ok. Now we usually know what depths we encounter. We have this knowledge of previous tracks when we encounter like a GPe usually around, usually in the same setup the same procedure and then same canual length and all of that. We know that about 60 62 63 we encounter usually GPe. And at about 50 starting at 50 all the way to 62 we usually have striatum. That's how you kind of previous knowledge that we have. We go by that and uh just try to keep that in mind when we encounter things. So um so there is a border cell at 61.8 that is still labeled as Striatum and uh then there

is at 62 a GPe Pauser and that is about right about where we expect it. The boundary between striatum and GPe is a little bit small it is only about .2 mm so cause striatum was at 61.8 62. so that border. That 61.8 might be a border that is part of GPe, rather than striatum. Most likely here. But that does not really matter that much. But anyway we got clear Pauser at 62. and at 62.3 a Burster 63 another Pauser Burster together. 63.3 another Burster. Injury at 63.1 he must have gone back up. By about half a mm. Or quarter mm. And uh 63.9 still a Pauser. 64.2 Pauser Injury. 65.1 Pauser. 65.3 Pauser, 66 a Border. 66.4 Pauser. 67.3 Border 68 uh. So 67.3 the last border and then at 68 we have GPi. So lets see we go from um 62 to 67.3 that is about 5.3 mm uhh of GPe that is quite the expected length. You know about 5mm is about what we expect there. When we look at the anatomy the general anatomy. So that is about right. So 68 for the GPi is a little bit high because we usually expect it to be at 70 but since the GPe started at 62 already and not 63 so that's a mm higher it is in the ball park so we'd say it is ok. I mean it is what. It is only a problem if that GPi is not a clear tonic a clearly identifiable tonic cell which is clearly different from a Pauser Burster from the GPe. If there is any question about this then it would be they would go through this thought process. But it is very clear tonic and very distinctive from the from the u GPe then it is clear that that is the start of the GPi. Uh. 63. is tonic and pausing now this causes a little a little uh question mark why there would be pausing there. Some GPi cells are known to have that sort of interruptions in their tonicity. And and Pausing is certainly fall into that category. If it was like really a pausing kind of pattern it would be called a Pauser and we would have to question whether these is still not GPe. There . 68.5 bursting pausing. Bursting now again is another um um label that we use for some of the GPi cells when they are uh either injured or have a unusual pattern. Bursting though the bursting that we call bursting is is distinctively different from a Burster itself in the pattern. The

Burster has this clear Burster discharge which is definitely not near bursting. So and uh 68.5 this is twice there. So they must have gone up, uh interesting it did not show that he went up he is just sitting there for 4min of course this was at 30 68.5 and then at 34.02 still at 68.5. so he must have at this official site. He must have recorded this site twice. Umm. I need to read the comments back there. Turn signal on, turn signal off. No response. Move arm. Ok. So now I he did at that site first site 68 he did a tonic cell he did a sensory motor exploration by moving the arm and got no response. That happens quite frequently that you don't get responses. Especially if you just move the arm this might be a cell that is related to trunk or leg or what have you. So. And at the next cell he did not do any exploration that is 68.3 and 68.5. now a bursting and pausing cell which he might a site also and recorded. It said jumped in numbering. That is not correct. I don't know what he meant by that. Whoever wrote that down. Jumped in numbering. Um. Sorry I can't make any sense of that. Ok. 68.5 still move the arm, up and down and says it looks like GPi. And then he moved the arm up and down again. I assume that is passive movement up and down. Usually we do also flexion and extension and wrist if we do an arm exploration and fingers. Just to see that. But he said no response to patient interaction, I guess pi. Alright um then he goes down to 69.2 and find some pausing cell again. Now pausing is is again not the pausing that I would expect. Pausers we do by the pausing that is some GPi cells do that and he tries the foot on this one um. No mark there whether there is a response, elbow flexion. Move arm, move leg, arm up down and I guess there is no indication whether there is response or not. So we don't know whether this responded or not. And no depth but it says border and chugging so I suppose there was no. depth was omitted on this one. 69.6 bursting again bursting type recorded as a site. Arm um to 25.3 ok. The time step does not make any sense. That cell was encountered at 49 uh 53 and then it says to

2:52.3 I guess they explored that from 49.53 to 53.23. 70.5 border cell and um 70.9 a chugging cell. This is a type of chugging discharge that is usually associated when we hear chugging we think about tremor. There could be some underlying tremor mechanism that is driving these cells. So at that point we say uh lets see whether there is some kind of tremor. Um in any of the extremities like indication like going like. The chugging rhythm is usually associated with 4-6 Hz. Tremor frequency. 71 this is tonic cell. 71.7 border 71.9 GPi again that is . I guess there is no this is all GPi. Once you have encountered GPi you pretty much stay in the GPi unless you get into the accessory lamina where there is some time quiet quiet uh quiet stretch there. You go through a stretch of GPi depending on where you are. Shorter or longer. But there is usually continuous structure GPi until the ventral border is encounter. 71.9 says GPi and then right after that it says quiet so we must have at 71.9 encountered a break which is very a very short run from this is from 68 through 72 well that is about 4 mm which is which is a small but significant stretch through GPi. That tells me for instance that it could either be lateral where the GPi kind of tapers of or we could uh be at and posterior edge or an anterior edge. Where the GPi tapers of so we get a shorter run. The long run would be right in smack in the center of GPi which can be up to 8mm so we get quiet at 70 about 72 and we encounter a border again which can happen. Quiet usual means that the background activity which is cause very strong when you get inside GPi background activity has tapered of so the background no clear cells but the background has quieted down a little bit that means you get to the edge of it. And you can sometimes encounter some border cells like you know intermixed there. Um then you get into 74 it is definitely quiet it tells me that we are out. 72 a border 74 definitely out of GPi at the ventral edge and then there is an injury so there must have been a last cell there so it might be exactly the border there. And that is what they did at 74.5 they started then, well actually

at 74 they already started shining the light activity. into in the eyes so they are testing the optic track. Uh since it was already quiet at 71.9 and they think, usually the optic track is about a mm to 1.5 mm below the ventral border that is why they are checking for light. Here and an injury this injury could have been a fiber. That is not necessarily we can not tell from an injury unless it is clearly a neuron discharge. Whether that is a fiber or a cell. So we are definitely out at about 74 they start to shine the light um. Mouth and speckles now it says in the comments lets try stimulation. Turn signal off ok. It must have been the old system where we actually deactivate the recording. We have to do that because we did not have an automatic switch to do between the stimulation and the recording so he probably turned that off. And uh and went into stimulation mode where you can here the click when he does the click. Um and so when he did stimulation he saw mouth twitching and speckles. Mouth means there is capsule side affect. Means we are affecting the internal capsule motor portion of it. And the speckles means that we also are affecting the optic track so we also stimulate the optic track which is the speckles of light. Which phosphenes so we and the stimulation with light says 40 um that I guess that is 40 mA so those are two different things. Stimulation light that is one thing the 40 is actually electrical stimulation which is 40 mA. So this a little bit cryptic here. And then it says optic track –ve and internal capsule –ve which does not make any sense at all because we said we had mouth and speckles which is indication of interaction if you saw something mouth affect like twitching of the tongue or corner of the mouth then it should be internal capsule +ve the plus so that's kind of inconsistent. 74.6 it says patient does not see speckles now. That is optic track –ve and at 75.5 stimulation light optic track +ve now. So we went a little bit further down and we got into optic track or we got close enough so that we get response. Um. There is a signal on now which doesn't make any sense, because during this procedure we would not turn the signal

back on. He would just unless he was checking, if you here a clicking sound he must have been checking the impedance of the electrode. Um which he does in between stimulation. Um there always consist stimulation degree you always want to make sure that the electrode is still intact or is still viable by checking the impedance in-between. Especially high intensity stimulation like 40 mA and higher. Stim light I am using he is shining the light in the eye with a flashlight. Optic track + at 75 light in eye 75.3 stimulation 50, which means like 50mA and uh there is no optic track in all this there is optic track minus –ve and internal capsule –ve. That means uh that high intensity, even at that high intensity what 50 mA um we did not get any. So were actually um. So far we have conflicting evidence between stimulation micro stimulation and light shining the light. Which can happen sometimes. Sometimes the um they don't deliver the same result stimulation and the light, stimulation with electrode sometimes they don't correspond or not confirm each other. Could be either one or the other or both but it does not always have to be together And then that is the end of depth at about 1hr 1 minute and 20 seconds which is a long relatively long track. Ok. There is the map this is of course for all of the rest of tracks. And it seems like we did in this case it looks like my handwriting . um. We did 1,2,3,4,5, 5 tracks and uh we did 3 4 lesions which are stacked. And uh is there something I was suppose to cell type classification, I talked about layer determination and um precautions taken throughout the procedure. I mean there was no precautions that were mentioned here. Well what we usually do is we watch out that the patient does not um go into seizure so we try to intercept that when that when that happens. We give some medication to to block that when the seizure happens that that means that is the end of the the cause usually it can take can last hours and then we would have to abort the surgery. Um those are the only precautions. Of course the anesthesiologist watches for all the other vital signs and make sure that

the blood pressure is is in limits and the heart rate and the breathing and so forth that all the vital signs are ok. And those are the precautions that we would take. Other than that we we watch for, when we pull out the electrode we watch for excessive heme on the electrode cause that means there could be a micro bleed, a micro bleeder there which could cause to some , if it is higher up some seizure activity. If the cortex gets irritated. So those are the procedures the precautions that we do. Um.

Ok now we move on to the our question 4, given the next two pages does this reinforce or change your belief findings from question 3. ok. Now what these are, these are actually lesion tracks that happen later, that happen after all the tracks have been completed. At this point in time I would not have this if I want if I am asked to go back to this track and see whether later on after all the tracks have been completed and we did the stimulation, in these areas which by the way the track one is in the center. Where we did lesion track number is 1 is here is is 1,2 3, 4, 4.5 mm back from track one which uh uh obviously uh obviously they chose these posterior location because track one was to anterior. So for not being the correct target. Looking at this here I would conclude that we did the following, we said ok, track one is probably anterior. Track 3 was to posterior, track 2 seemed to be uh 1,2,3, 4 behind track 1 and 1,2,3, in front of track 3. so that seemed to be the most appropriate place for lesion for our most lateral posterior lesion. That is the conclusion I have from this map. So it will lesion very close to track 2 and uh and that one when I look at the results. The depth this is now macro stimulation that we do lesions at 300 Hz and uh I assume this was 1 mA and we found some capsule response at 1 mA at 69 a depth of 69. we move down to 70 about an index finger of about .7 mA and uh then 71 we got .6 mA and at index finger. So mostly fingers. Now that is an interesting uh finding that it means we are actually more hand arms usually

lateral. So this is our most lateral extend that we would put a lead or lesion in. we don't want to go any further lateral than that. And as I see track 4 was two mm lateral from this current precision and that tells me that that is what they based their their findings on or their presisioning of the first lesion track. They said ok 4 was probably much to lateral. 2 was good so lesion track confirmed 72 are edge was the border was quiet 74 72 there is about a ventral as you want to go. Other wise you get out of the GPi and into the internal capsule or optic track. And we test this with a threshold at 72 73 of 0.4 mA. That is a little bit low, we want to stay at .5 mA uh for responses and above. Anything below that gets to close to the structure. Now 73 equals 0.4 that is to close so we back up to 72.5 and we still have 0.4 mA of capsule. So we back up another ½ mm to 72 and when it was 0.5 mA which were pleased with and that is kind of what we stated depth and then we also tested the optic the visual and that threshold was greater than 1.4 greater than . for the optic track we usually have a threshold of 1mA and above. Anything below 1mA is too close. So that depth of 72 for the second stimulation they were pleased with that so they actually started lesioning. At a depth of 72 at 60 degrees which is usually at the bottom we usually make a smaller lesion not a larger. Which is 60 degree. Then we observed an arm dyskonesia a slight dyskonesia and arm dyskonesia which sometimes happens with the coagulations. You sometimes get dyskonesia which it is actually a good thing. It means we are actually effecting the area that is diseased or that is has the defect. The bradykenesia tremor or what have you. So that is a good sign, the dyskonesia is a good sign. So then we set ok 72 since we got this dyskonesia we will expand it to 70 degrees Celsius which makes it a little bit bigger. Then we backed up a mm higher up in the track at 71 we did a 75 degree lesion. And then I assume that at 70 there is another 75 degree lesion. So we stack them dorsal ventrally so that expand up into the whole extend of the the nucleus in the GPi. Ok so confirming I would say since

this is not the right since that it is 4.5 mm away from that um now I would say that the short extend of the GPi that it was a shorter run through it can be because it is anterior where it is tapering of or posterior where it is tapering of. So the shape like this. Anterior you get less of a run through it. And when you go posterior you get less of a run through it. So that sort of confirms that this track was anterior. Um the striatum was a little sketchy here if there was a run. If assume it was continuous from 50 to 61 that's about 11mm . 12 mm of Striatum which is a fairly good run. If it is continuous it means that it is either anterior which we confirmed with what we just said or it is lateral which is clearly not the case. Um so it was anterior. That goes together with that. Short run through GPi and long run through Striatum uh tells me that 1 was anterior. And that is kind of what the rest of the tracks tell us here so. Ok. And looking at the lesion sheet. I see that they stacked them as I said they stacked them down 1,2,3 lesions 1 mm higher up that gives a a longer extend of a lesion you cover all the. Cause each lesion is about 3mm in diameter. So if you have more than that of GPi you have to stack them up to cover all the rest of the depth of the of the GPi.

Moving on to question 5. what activities are involved in deciding where the next track should be placed? Ok what we so when you take the first track. And as I said you discover that the large running through the GPi to the Striatum. Large run through the striatum as I said indicator either lateral or anterior. Um you get a short run through, now this is difficult. You get a short run through GPi. Also um. That could also mean that you are lateral. Long run striatum short run through GPi. Normal through GPe could either mean that you are anterior or lateral. So at this point at 1 it was actually. They made a decisions. It was actually not clear whether to go lateral , whether we were lateral or not, or anterior. Uh the things is that. There is a golden rule that we have we

stay in the plane that we are mapping. That is the golden rule. Don't leave the plane. The Parasadual plane unless you don't get any more information out of it. So we try to move anterior posterior this is our golden rule. And so we assume that we could either later or anterior. So what we do then is we go posterior. Cause if we are anterior. And we go even more anterior that does not make any sense. If you are lateral it does not matter if you go anterior or posterior to see how much we have in front or behind it. Uh so I what the thinking is right now is at this point ok if we are anterior we need to go posterior. If we are lateral does not really matter if we go anterior or posterior. So lets just based on these two go posterior and this is actually what we did so we went 1,2,3,4 posterior from our first track. We usually take a 4mm step if we really think we are anterior. That is if we close to where we want to be and we want to find the posterior edge of GPi we go only 2mm step. Cause if we are already in the middle of the GPi you don't want to go totally out of it and not find anything. What our plan is every time we run a track we want to find GPi. Cause if you don't get GPi then that is a negative result and we don't know whether we don't get GPi because something is wrong, we have a bleed you know or whether we are out of GPi. So it is always better to find GPi then to not find GPi. So um um so were going 4 mm posterior and in fact we have I don't have the data for that and we probably actually got a nice long run. Actually I got I got a data sheet here, ya we got a nice long run about 1,2,3,4,5 about 5 mm which is uh not overwhelmingly long but is long enough to have solid GPi there and for as another result is since were totally posterior and still got a nice run through GPi it definitely will not be lateral. That is the conclusion from that because the extend of GPi lateral. If you got a short run then you would be in the center a run like what we had 4mm you think only get that lateral if you are right smack in the center. If you go 4mm posterior you will lose it all together so that is the rational. So this tells me right there 4mm posterior you should get

a 5mm run through it, we were actually are medial but the first track was also to anterior. So that the next step would be then ok lets find the posterior edge since we still found GPi there, lets find the posterior edge because we want to know where the edge of it is where the capsule is because we want to get into that. We want to stay out of the capsule so we want to know where the posterior edge is. We don't want to lesion or stimulate to close or DBS to close to the edge the internal capsule it is just trouble. So they went 1,2,3 back from track number 2 which is a little far. I would not have stepped. I would have stepped 2mm back but they did, and sure enough they did still get some lets see 1,2, mm of GPi so I guess this is a long a pretty big big anterior posterior GPi because we got GPi over 1,2,3,4,5,6,7 mm which is about the largest extend you can have there from anterior to posterior. So that seems to be a very short run the posterior edge there. The posterior edge would be right there, where track number 3 is, so you want to stay away. That why they said ok lesion two when we have a posterior edge we want to stay with a lesion we want to stay away 2 2.5 mm away from that edge. And that is what they actually did and so 2.5 they put the lesion 2.5 away from track 3 anterior from that. So um so that is the things we consider then how much a run we have through GPi um is it um anterior then we need to go posterior is it lateral um then we need to go medial. If we are to lateral we don't want to get into GPe, if you lesion to lateral then you lesion part of GPe which might not be good. Um if you are to posterior you lesion the capsule and that is not good either. If you are to ventral you lesion the optic track that's not good. So so these are these boundaries the lateral posterior ventral boundary that you want to stay away from. And then anterior that is ok. You can lesion there. So they also explored with track number 5 medially, and uh that is done to make sure that there is plenty of GPi there to move of course this even though the track 4 was lateral to lateral track 1,2,3, where on the lateral side lets put it that way. You don't want to go any lateral

than that. You certainly if you want to whole whole GPi you want to be more medial for some real lesions and track 1,2,3 and lesion 1,2, that s why they went and recorded in-between sort of anterior from track 2 by 1mm and medial by 2 another track 5 and I guess they found that there was still plenty of GPi there. Um. So they placed a lesion close to that. That is the third lesion and then anterior of that for lesion number 4 in the same Parasadual plane but 2.5 mm anterior of that. 2.5mm in-between lesions is usually our common thing because then our lesions overlap anterior posterior by a certain small amount. So that's that the strategy for a lesion. Now this was done in 96 so that was rather early on. I think that when they do lesions now they leave the anterior lesion off. They we have discovered now that maybe going to more medial to anterior we leave for dissociative an uh and uh areas little bit of areas and we don't know if this is really beneficial. This is still a question on whether this is good. They do more of an L shape kind of lesion these days. So they uh have just 3 and 4th one anterior one and medial anterior they leave off. So um and I mean I think I explained it enough what our strategies are to decide where the next track should be there.

Uh question number 6. what would be your recommendation for where the next track should be placed with the information provided in question 3. I think I explained already also. That uh you know that I would stay in the plane and would go posterior that's the exactly the strategy that I did. And um this is some maps where they actually placed we placed the placed actually where we think we mapped and it did it does confirm. The readings here so. That is pretty precisely there. Um.

Ok question 7 anything else that you would like to add regarding track execution or procedure. Yeah I was just getting into this this we draw the map the sometimes during

the procedure I have been doing this to see where we actually are. When we do the targeting the targeting was lateral 21.5 and uh according to the surgeon and anterior 2. anterior 2 means off the mid commissural point. Um the these readings are estimates they then we try with the mapping to reposition this actually where we actually are according to the according to the findings. So this is a tool now with the new procedure this is the same thing we would just all of this on a computer graphically rather than doing it on a piece of paper. But this is a good tool to see actually how it makes sense how much these tracks made actual sense putting it on the contours um. Ok that's that.

Task 3

So this is the task three follow up questions. Um ok same thing as the explaining reasoning and answer for the question. Question 1a how do you determine when you have crossed into a new anatomical layer? Well one thing is between layers there is there is a steep silence or quiet cause you only have background noise because there is no grey matter there is no neurons around uh in close by the electrode tip. So when you get a quiet and then all of a sudden the background activity increase you get into the new grey matter zone um that's that is usually and indicator there. Drop in background activity increase in background activity so that's the good indicator. If that is not that obvious then of course you listen to the neuronal discharge and if there is a quiet or no neural discharge or all of a sudden a totally different neuronal discharge like striatum is distinctively different from GPe, Bursters and Pausers so uh the distinction between GPe and GPi is not that clear at times but you still get quiescent areas period and there is a lamina of about half a mm to a mm that you expect that to happen and you also know how long the layers are, what stretch you have for instance the GPe usually run 5mm and Striatum can be as long as in the previous example of about 12mm. Um so neuronal discharge, quiescent areas different types of neuronal discharge all of a sudden those are determinants.

Um 1b. what factors do you include when deciding which anatomical layer the tip of the probe is located in? um well factors are pattern of discharge. The pattern of the discharge is is temporal pattern of the discharges is very strong. It's the strongest indicator. And um and we just have our hearing we are trained to distinguish between these different types of discharge whether it is a border cell versus a Burster, Pauser

GPi tonic, GPi or Striatul discharge which is like popcorn that is a very distinct discharge. So this is like a temporal pattern of the discharge. Uh

Question 2 a what kind of uncertainties exist in determining the probe location in a given track? The z-depth in parenthesis. You may list the answers as you talk aloud. Well uh the uncertainties uh as far as the depth reading is of course very precise that is the error there is very small so there is no physical equipment uncertainties that um linearity . um the in terms of as I said the most difficult distinction is between GPe and GPi if there is a pausing kind of discharge in the GPe and the GPi compared to a Pauser in the GPe, that can sometimes be difficult to distinguish between a temporal pattern and uh so there is a uncertainty. The other thing is all of a sudden when you lose a signal um you don't have you don't expect it to be over yet you don't expect to be out of the nucleus yet and all of sudden you use it and you have a stretch of 1mm where you have no discharge. That is a a concerning thing. If it come back then its find, but if it does not come back then its get quiet and in background discharge we assume there is some kind of a hemorrhaging going kind of some bleeding . so that will knock out the recording. We will just say that the electrode somehow got damaged and we do testing of the of the electrode impedance. But uh in terms of uncertainties um between other types of discharges there is not much . cause the patterns are so nicely distinct from each other. Unlike other areas of the brain where they all sound alike. So that is actually why this mapping works that well. Um I cant think of any other uncertainties of probe location.

Question 2b to what extend to a small or large or large degree do these uncertainties exist in the determination of the probe location in a given track? Um lets say the the biggest uncertainty is of course is where whether you are in the GPi already or whether

you are in GPe and one uncertainness is that if you are to lateral you are running through a long stretch GPe and the GPi is not over at 5mm but keeps going on and then dropping out and uh it sounds like a Pauser pausing and you don't know are you in GPi yet or are you still in GPe because you are lateral. The biggest uncertainty then is that if you have a long stretch of Pauser pausing like uh discharge that you might be in GPe lateral and you might have missed GPi. Other than that I don't think there is any other great uncertainty. The other things is also when you go anterior very anterior you can run through a longer stretch of 5mm of GPe and uh and miss GPi altogether. And then uh so as I said before. If you don't find any GPi that is always bad. Because that is you know you don't know did you miss it where did you miss it to anterior to lateral and you don't know if your electrode works right do you have a micro bleed there and hemorrhaging and so no result is always bad.

Um what kinds of uncertainties exist in determining the track placement? Three dimensional mapping in parenthesis. You may list the answers and talk aloud. Um the uncertainties are the resolution of the MRI so the that's one thing I mean you make the decision on where you run the first track where you target the structures by looking at the MRI. And then actually pointing there and saying this is where I want to go. Um that's what we do now with the stealth. The so the resolution of the MRI the definition of the structures um how the neurosurgeon determines where he is on the MRI that that's the thing. And the other things of course there is a small variation in couple of mm in each direction of the targeting apparatus itself. It could be some kind of um reading error on the transfer of coordinates from the stealth on to the actual frame setup. So it's a mechanical. There could be something there and uh so those are things that are actually um basically determining how well you targeting the first track. And um and the

to what extend small or large. This is a large these errors with MRI can be 4mm which is about half the size of the target. So you can be off either miss it anterior or miss it posteriorly um so it is a it is a substantial error that you can do. And you have to adjust for that. Um but it is not an error that would totally throw you of or you know get you lost in the brain. So um that is not a such a great error. A couple of mm error uncertainty in the equipment and the and the MRI together can actually together work to give you a composite error that you might miss GPi anteriorly or might miss it posteriorly. But you still get some kind of striatum. You wont be totally away so you get lost. Uh ok … I think I answered the question 2d to an extent do these uncertainties exist in the determination of the track. That is meant to actually what extent. Um every time you do that you run into that. Every time um you do the targeting you have the same problem. So there is o learning or adjustment possible of the equipment. So that error never changes. And there is some medium kind of error it is not a large extent.

So question 3 how do you determine if and when patient interaction is needed? We um always try and do it. We and we always try to determine it while we are in the GPi. You don't test in striatum you don't test in GPe. Rarely. I mean sometimes out of fear, out of curiosity you do GPe it is not important because the sensory motor response you get in the GPe is different is not aligned in our Parasadudal plane with the sensory motor map of the GPi. The the corresponding map is slight ant an angle over so you have to be more lateral in the GPe to get the same response as you would in the GPi for a more medial spot. In GPi. So if you are in that parasadual plane your response is that you get in GPe you don't have anything that project to the GPi. But we try to do it every time we have a cell in GPi and only out of time constraint we don't test every cell we find in GPi. But we try to find at least a couple of good cells that respond well to sensory motor

exploration. And uh to tell us what some sensory section we are in and whether our lesion DBS implantation would have a adjustifiable or a predictable outcome. Um

Question 4a what types of patient interactions exist and when do you use each? You may list the answers as you talk aloud. Do you want me to write it down or or I think its just as well documented here. Uh the patient interaction is a) passive movement of the extremities which is determines whether we are in the sensory motor area. We also ask the patient to do active movements. We do it on both ipsy and controlatteral side. And uh um we also probe the back because they cant move their back. So we just probe it the perispinal muscles to see if we get some trunk there. Those are the the sensory motor part of the exploration and the patient has a tremor and we find a tremor relation. A tremor modulation of this neuronal discharge. We ask the patient to hold up the arm , to let the tremor set in. if the tremor does not set in then we make the patient count backwards in 7 or something like that. Which puts a mental stress on them and uh and then they uh that kind of throws them off and it makes the tremor comes out a lot stronger. So then we want to listen how the tremor is related to the neuronal discharge. If the it sounds chugging like ch..ch…ch…ch…ch…ch like this you see it in relation to the actual tremor. Um synchronized with that. Uh The other uh thing that we do is the optic track and the internal capsule. Uh which is the micro stimulation and the uh multi track exploration with the light. And we ask the patient um for one thing the with light exploration of the with shining light in the eyes purely passive, the patient has to do nothing just keep their eyes open. And uh look into the light so that we can have the most biggest light affect. Um this micro stimulation we have to for light of phosphenes we have to ask the patient whether they see these speckles or bright lights or colors or what have you. And then we need some answer from them from the patient. For the

capsule this is again strictly passive non patient there is no requirement for the patient to cooperate there except that we want them to relax , not to contract the muscles. And then we watch the affect passively the micro stimulation whatever it does whether it is a twitching of the mouth the tongue, the side of the mouth, facial where there is some twitching in the extremities, some twitching. So those are the micro stimulation affects for capsule and then light, I mean an optic track. And then the light. So that is all of it. And the uncertainties with these factors um well uncertainties is sometimes when the patient has a tremor facial tremor it is hard to see whether we actually get a twitch of the tongue, the side of the tongue, and get some really minute or or borderline threshold effects from micro stimulation. That s what capsule. The light the phosphenes the so called speckles they either see it or they do not. But we still do test and do a zero turn it to zero and stimulate and say now…now…now… and you test for the patient actually they should not actually be seeing anything. If they say they see it then uh we know that they made the other stuff up to or that they don't know what they are actually suppose to look for. Um. And uh we do that test, we turn the intensity down to zero. We do a couple of zeros sometimes if we are not sure that the patient is actually aware and alert and uh. Of course for all of these procedures we need the response from the patient we need them awake alert patient. Not someone who is drifting of to sleep or you know getting delusional. So that is the that is that. So uh So there is uncertainty factors with those. But other than that if you stimulate with the uh with light and you listen to neuro signal. Sometimes the optic track discharges is are slightly subtle and uh so that is an uncertainty exactly where do you get the response. But if you get the response and you heard it once you can reproduce it and hear it again and again. So it is just a threshold thing. If you are not close enough then some people can hear it others can not. So um that is a little bit more subtle so the optic track with light is a little more subtle.

Stimulation is much more it is either there or not you either get the response or not. Uhm unless the patient makes it up. Of course there no way to tell whether the patient is sorry or not it is something that is like psychophysics you either know it or not. Uhm only by believing the patient. So um

When listening to the audio signal during the track how do you determine what cell type you hear? Uh as I said this is the learning the experience. You got this pattern that you learn, the neuronal discharge pattern and uh and over time you get to distinguish between the different pattern types. And as I said there is only a couple of them that are a little bit less clear. Not so tonic GPi versus Pausers in the GPe. Those are the only difficult ones. The other are pretty distinct from each other. Um. You the question is can associate the question is can you associate cell type with layer if so what are the associations? Clearly the cell type in the striatum is either what we call caudate Putamen cell and the other clearly distinguished cell is the popcorn. So those are the 2 for striatum. Uh there is also some border cells which are the edges of the structure. Uhm the GPe Bursters and Pausers and borders. Borders are very tonic slowly discharging cells which are always distinct uh most of them. Some of them are can be a little bit faster. But we still determine them as borders. And the tonic cells and chugging and bursting and pausing is in the GPi but they are usually discharge. And I should say temporal patterns aren't only patterns it is also frequency. Of discharge. How fast is the discharge. And I should have maybe mentioned that earlier. Uhm the higher the frequency the more likely it is going to be in the GPi so if you have the distinction to make between the GPe and the GPi and uhm it is a very high frequency you are most likely by that high by that higher frequency say that this is more than likely GPi than GPe. So in general the frequency is in GPi is higher than in GPe. What cell types are

difficult to discern? Identify listening. Identify when you listen to the audio signal. Ok slash fine. As I said Pauser versus like tonic GPi sometimes there is the biggest overlap the others are clearly distinct form each other.

So what tools are used in determining where to place the next track and what information from the current track do you use in making this decision? Um well the tools that we have right now, other than the computer program. We draw a track the extent of each nucleus at each, essentially the white grey matter distribution along the track. We associate certain um stretches of that track with certain areas of nuclei in the brain. And we take put it on a clear transparency um which is a whatever, transparency. And uh so uhm then we move this over unto a scaled um template with contours and try to match this on there. Um we have a pretty much an idea of the vicinity we are in that's we we already pick a preselected map and we try and place it on there. Only if this does not make it any sense at all it does not really fit we go to the next layers, go up and down and try to fit it on there. So once we found a match that looks pretty good in terms of all the extent s of the nuclei we we encounter we say ok this is where we are and then we remember all these other maps. In relation to that where would you go in that same parasadual plane to get the next you know most information out so lets say you where anterior and you want to go posterior. You want to go posterior not to much that in case you you are not where you think you are that you would loose GPi and not get any at all but it don't make your step that large that you step out of GPi so to speak. You make it large enough so that you get to the posterior edge of the get enough information out of it so that you don't need to make another track in that same plane. So it is kind of optimize the next step. Um it is just like an economical kind of thing. Uh and then uh and that's the decision that you make. As I said the golden rule is stay in the parasadual plane that

you are don't step out of it, don't make two steps, two changes at one time just stay in that plane. This is mostly true because we having this Parasadudal maps. If we had any kind of map that we wanted, that we have in the computer that you can actually re-slice it you can actually step sideways as well if we think that is the most economical thing to do. But we could We don't have to stay in that parasadual plane. Mostly we do that because this golden rule has proven itself because we have these parasadual contours. And you want to explore that contour first. Um

When performing subsequent tracks what information do you use from previous tracks in to understand the current track? Um we do when you go down for instance certain boundaries are gonna occur at certain depth corresponding to the other track. You would expect like GPe to start at a certain depth. And the variation between the two is not gonna be large because the curvature of the GPe is almost like a straight like a straight line, so when you come down in another track down 2,3 mm posterior you expect GPe to start at about the same depth. And so that is what you use form the others. You ask where did I find GPe where did I find GPi and correspondence to that uh I would uh already be looking be on alert to look for the for the beginning of these nuclei corresponding to the other track I so that gives a indication. You never now how far actually they put the aperture the the gicanular In terms of the surface of the brain. So you know how deep you are in there. So it is all relative. We don't right now have a fixed starting point that we say we always start in a fixed relation to the surface of the brain. And then also the cortex could be higher. We do not start lets say at a certain fixed location that always gives us the occurrence of GPe GPi in the same depth. Can vary quite a bit like 4 5 mm up and down. And so once we have the first track. That gives us that nails it down. Now if we assume that we are at this laterality if we go down another

track 2 mm behind that uh in front or behind that we will encounter these layers at certain depth. So that is what we use for information there. So I mean what this thinking is always look for the occurrence of these layers, it does not matter each track in itself is unique. You can use some of the information but the performance of it fact it perform of the track. Run the track down and map whatever you find. But it gives you an indication when to be really alert when to really look for things.

Question 7a under what conditions do you pause the procedure and what actions do you perform? Um yeah this is when the patient is in trouble you know as to go to the bathroom or something like that you know. Um when there is some problem with the patient having some kind of discomfort on the pins that are sitting in there head there. True and get some anesthetics in there. Um when there is a bleed, when there is an excessive leakage of CSV or the any kinds of condition of the patient that requires your attention. So um and of course we stop between the tracks and we have to clean the canule and clean there electrode if there is possible heme on there uh so in-between the tracks is where we actually stop and we have to pull everything out. And service it to some extend. And adjust of course the x, y station to move to the next plane .to the next track location. Uhm and uh of course if the electrode does not perform, or more if the impedance has dropped if we have any question of about the workability of the equipment then we stop and examine that. Um that is if we do not get the expected recording. Especially in following tracks. You know tracks one you don't know what to expect yet, we have an idea but anything goes you know, in fact track 2 is to a certain point. You know you would get this and this layers. If you don't get it you kind of have to stop and wonder what is going on. Sometimes there is brain shift occurring. That is another thing you guys have probably not taken into account yet. Brain shift means that

the brain shift downwards and backwards due to gravity and CSV loss and then of course you are in trouble and you have to start mapping again because you lost your bearing in the brain. And all the facts previously, unless you know exactly how the brain shifted and where it moved to this uh you can forget about it. So you have to start over, track 1. So but this is not a big deal, I mean it is a big deal for the surgery, but it is not a if there is enough time left you can just start over and remap. But those are things that we would stop the procedure. Stopping the procedure is another thing and if there is a seizure of course then you have to you know stop the procedure if the patient does not recover from it in a reasonable amount of time then you just call it. That is question number 7b. of course if seizure and of course if the patient is not otherwise in trouble and there is severe serious complication in to some the condition like heart or what have you. Um that when we would abort the procedure. Under all circumstance we try to finish it. Uh as best as we can but we get a lot a there are certain times when we just have to say you know. We would restart the procedure uh if the patient has not have a a higher seizure and it comes out of you know and the post dictal state like in 20 minutes or 30 minutes or so. If the patient is alert again and most and everything looks relatively normal. We would then start another track. Start over. During the seizure of course the brain could have shifted of course and so after threat we run the next track in correspondence to the

Previous tracks or track. But if we will very weary about that. And if things don't match until it can make sense, we consider these previous tracks as as non void for the range of procedure.

Question 8 in the ideal world what kind of technological assistance technological assistance would you like to see to improve the process. Um better MRI. Better

anatomical definition of the structures to target better. Uh the uh away to um monitor the probe better in real time. Where you actually are in correspondence to the structures. I mean if we had interventional um MRI or imaging procedures any in the or that would be a plus, fantastic. Uh that it could confirm where we actually are to the anatomical structures at any time. Um with the probe, but the probe precision in relation to it. Um uh other than that um assistance in term of previous lesions I mean procedures on the patient. If we had the information of that and could superimpose it onto the current um that would be something that is a real plus. You know we would do the one side and do the other side and you would already have the information of the first side. Things are usually pretty symmetrical in the brain. So you could essentially draw some conclusions from the first one to the second one. Um and uh other than that I think technologically the apparatus for mapping is better. Has a high precision that would be most preferable but since it is only a couple of mm it is not that critical. But the MRI is a little bit more critical.

So question 9 how do you anticipate… well of course the other thing we were mention is that we have better imaging, better representation of the models of the actual patient specific anatomy. That is the other thing. If we don't have interventional MRI and mapping anatomical imagining during the procedure the best thing we can then do is have models that can simulate where we are going during the procedure. So we can see the approach and have a model of the brain structure in that are that is patient specific not just generic maps that is really patient specific anatomy and we can see where the probe goes through. If we go here or there or there what will we encounter. And also what we encounter at the current position. That is something that uhm that would be great. And we are working on that.

This technology, how do we anticipate using this technology? Uh in track mapping. We would of course monitor with the real life time real simulation where we are going in the track with the model. And would give us an idea where we are encountering, would be encountering certain things, certain structures and also to make planning once we have confirmed this is we here it is. Could actually already stop there and say ok we need one track and say this is our current location. What would be our most optimum location for the lesion or DBS lead implantation. Could actually do that based on that. Um we would know where the structures are that we want to avoid due to the model and know the structures that we really want to be in and want in to affect. And that would be the ultimate goal.

Question number 10 now that have completed both tasks, is there now anything that you would like to include that should be captured by our system? Uh well we haven't talked about the guidance during the planning of the track and the guidance during the what you are trying to do during the lesion, DBS placement. Could model that before hand if you know, if you had a good representation of the anatomy. You could already pick the ahead of time the places where you want to put the lead. Or the places where you want to put the leads or the the lesions. So um the lesion planning or DBS lead planning uhm will be something. You could consider and um I guess if you want to consider the patient the success of the patient later on the procedure I mean not the patient himself. You know how well perform these tasks that we have. Expert 2 can tell you more about this. When we do a scale of determine how well the procedure actually worked on the patient. That is something that is um would be nice, it is already. There is already databases available for that, um where they enter these tests you know. Test results and go about. That is something that we could possibility include we could also have the patient

perform certain tasks during the surgery which we also do sometimes just to see the um how affective certain things are stimulation during the operation to do sensory motor exploration complimented of course by stimulation of the regiment. Anyway that is something. I am just getting side tracked there.

It says discussion, would you like to discuss any of this or do you have any.

Baker; you had mentioned something about if there is enough time to finish the surgery. Is there a maximum length that you look at or what do you measure time by? Something else scheduled in the or.

Klaus; look it is discomfort of the patient expresses or he grouchiness of the patient and the cooperativeness of the patient. I mean if the patient depending on age and condition they get tired and if you do a lesion for instance. And you need the patients cooperation you can not wait to long otherwise the patient loose is totally uncooperative and you don't get the response out of them and you don't know if you are close optic track or not the patient does not respond. Of course the risk is if you go to long that the the you have risk of side affect, adverse affect during the surgery occurring. Like stroke or seizure what have you those are the limiting factors, we sometimes go 12 hours for a surgery. So it can take very long but it shouldn't and it is very very uncomfortable for the patient to be in that head frame. They get in at 6 in the morning or 7 clock in the morning and uh already so it is very uncomfortable to tolerate that and then they have to sit in that one position for like hours at a time. At times their back get sores they complain a lot sometimes more than others so and if they complain then of course they don't pay attention to what you to what you want them to pay attention to. So that is the limiting

factors. Other than that there I mean there is no there is like to go on with the procedure. The shorter the better.

Knowledge Engineer 1:

Um on question two I was wondering if could maybe just explain a little bit better uhm what factors are involved in you know at the border. How do you exactly well “exactly” using. How do you determine where the border occurs. What are all the different things that you do and all the different things that you take into consideration?

Expert 1:

Well with the border with reference to the structure?

Knowledge Engineer 1:

Like the border of the striatum, between the striatum and the lamina?

Expert 1:

Well the things is that as I said the base indication is that you listen to the neur0 the discharge. Whatever comes out of the speaker of the amplifier you hear that sound it is either louder or not so loud. There is more activity there and more noise there so to speak and less and whenever we get a certain constant background activity level and you get change in that background activity level like up or down, higher or lower in intensity or loudness then we will say that something has changed. So um when you get out of the nucleus, assuming that you get a constant background activity there, at times you find a neuron which is overlaid on there but then usually its like a background hash. And all of a sudden that drops and you hear that the ear can pick that up that means that

you are actually getting away from all these surrounding neurons all these grey matter and into white matter and there are usually no cell bodies no neurons that you record from. And also recorded fiber, most likely that is dropped in background activity. So when that occurs you know that you are out of the grey matter you are in white matter zone. And then when you get close to the next grey matter layer if you are increasing in background activity again and as you get into it there are more cells that are surrounding it you know not close enough to pick up real great spikes but close enough to pick up all the firing discharges and then to hash background activity level it goes background up. That is actually what we are working on now I am trying to get Dinal and and Quang to come up with that. So that they will actually get rms root mean square of that of that background activity and when there is an percentage increase or a decrease from the previous level then we will say something has changed. So we will detect that by measuring the rms or the peak the peak discharge levels. But that is but that is the main determinant. And of course if you get a clear cell isolated then it is even clearer and you hear it all of a sudden, you got a cell you are in the in grey matter. Fiber spike, inverted spike you know you are in white matter, fiber bundles. But that is it.

Knowledge Engineer 2:

ok can we take the tape and distribute to Vince if he wants to see it or Maribeth or john.

Expert 1:

Sure if they can stand me talking for that long.

B.2 Interview with Expert 2

Knowledge Engineer 2:

Ok. Let's go ahead and this is task 1. I f you want to use your desk that is fine.

Expert 2:

Alright. Please read the following instructions aloud. Please begin and remember to talk aloud. Direction. In this knowledge acquisition we are interested in what you say to yourself as you perform a task that we give you. In order to do this we will ask you to talk aloud as you work on the problems. What I mean by talk aloud is that I want you to say out loud everything that you say to yourself silently. Just act as if you where alone in a room, speaking to yourself or as if you where explaining things to to a group of residence. If you are silent for a length of time I will remind you to keep talking. Do you understand what I mean? Yes. Before we turn to the Pallidotomy knowledge acquisition we will start with a practice problem. Lets begin. I want you to talk aloud while you do this problem.

I would like you to solve the um anagram. It is your task to find several words that consists of some or all of the presented letters. Example if you if the scramble letter are k r o you may see that they spell the word rook. Any questions? Please talk while you solve the following anagram. So I just have take these letters and turn them around…

Knowledge Engineer 2:

You can turn them around to what ever words, just talk aloud while you do it, what are you thinking about.

Expert 2:

At first I have thought about science, but that does not fit. Um. I was trying think about how I would move the the vowels around if that fit a letter. I have never been very good at these. Lets us n e. I am just trying to solve the anagram and just talk aloud? This is interesting what is the reason for that?

Knowledge Engineer 2:

By talking aloud we are trying to catch your thought processes.

Expert 2:

Alright. That relates to solving a Pallidotomy correct.

Knowledge Engineer 2:

Well it gets you to talking aloud because not everyone is used to doing that. Now you tend to do it a lot in the or and so I think that you are going to be much better at this than some people. But to follow formal protocols we wanted to do this anyway.

Expert 2:

Well, I want to right them down.

Knowledge Engineer 2:

You want a pen?

Expert 2:

Yeah. Alright lets see. Well lets see I got an S C.

Knowledge Engineer 2:

Well you can right on that sheet to that is fine.

Expert 2:

That's ok I'll just use a blank piece. I am going to. I want to solve S T N and then go I O E to separate the consonants from the vowels. And then go lets see. I would try to stick in a vowel after a consonant. So I am just adding them in and kinda rearranging them. Sim cow and now. That does not look very good. Could it be sc something there. Like scien obviously but that goes right back to what you have. Never mind. Questions would be would it be a vowel first, not usually sometimes. I am going to start with an N. I am going to start keeping the same, keep the same vowel and I'll change the consonants. Take a different consonant this time. NI now go CI lets see CI . CIS , NI and then the other one S. ok. The best thing is to sit back and think about it, but I suppose I got to talk aloud while I do it. And it is hard to think when your talking aloud sometimes.

Knowledge Engineer 2:

And that is why we have this as a practice.

Expert 2:

Is that right. Yeah. I would want to use lets see s than o.

Knowledge Engineer 2:

And remember you don't have to use all the letters. It is some or all.

Expert 2:

Oh that's right, you said that. So you get the directions right. So if I just use some letters. Ok. Ok what I'll in this there are not multiple words to this I assume there is just probably one.

Knowledge Engineer 2:

Oh no there is more than one.

Expert 2:

So then you could obviously go sin, that would be one. So lets just try solving a couple. Sin. Nice, nice c e that would be a second one. I guess I am going to have to use wrong. So lets what about starting with the c or something. C I o, c o ce cem doesn't sounds so right uh what else could I do. I could take uh take the ne maybe, n just no, c anything else, do you want me to keep going.

Knowledge Engineer 2:

No I think that is pretty good. I mean the idea here is that you are talking aloud.

Task 2

Expert 2:

Please read the following instructions. Please begin. Remember to speak aloud. please explain your reason and answer each of these questions.

Question 1. During a Pallidotomy operation what is your overall strategy for a DBS or lesion. I assume for placing a DBS or lesion. Uhm the strategy. Number 1 is to identify the target, number 2 once you identify the target you identify where you are in the target, identify where within the target. Within that you identify borders, within that you identify the dorsal ventral border. If you are in GPi you identify also the posterior border you identify the lateral border. In STN you identify the anterior border, this is our approach. And you identify the lateral border. And then you would make. Ok so that is borders and then within that they will also identify nearby critical structures. You want to avoid. If you are in GPi again that would be internal capsule and optic track. If you are in STN it is not so much that you are identifying internal capsule or other places. You are really looking at finding the borders of that. Uh if you are way to medial if you found a couple of border stops and if you found capsule you are obviously lateral. It is nice to find capsule but I don't insist on that, if I do it. I just want to find the anterior lateral border. So that is my overall strategy. Identify the target, identify where you are within the target, identify the borders of the target, and identify nearby critical structures. Once you have done that, then you would place your lesion . and the strategy of placing the lesion in GPi is to first. Lets so lesioning, GPi lesioning in particular. First lesion. We make a series of three lesions in a triangle so that we get the posterial lateral portion of GPi, dorsal to ventral extent. So you first lesion is placement around 21.5, uh if you do it at

21.5 you will lesion a spread laterally to about 23 and coming medial to about 20. put your lesion back far enough, excuse me, so that you get internal capsule responses at the lowest point at a threshold of about .5 .6 .7 somewhere in that range. Look I got to remember, we have not done it for so long. I thing we are talking about .7 mA with our current system. That may vary based on different systems. Uh if you get .7 mA or .6 mA uh at that point you are safe to make a lesions at 60 degrees and you heat up to 70 degrees all the time monitoring the patient, to see if there is any change in their overall status if there is not you can heat it up to 77, lowest lesion. Come up 2 mm, 1 or 2mm you make another lesion it may be hotter there cause you are farther away 75 maybe 80 in some cases. I know some places make it as high as 85 90 we don't do that we do 75 80 at best. And then move it up to the extent of what you GPi would be. So if you saw a 5 mm run and that location, you know its about 5mm. Once you have done all that you move forward 2 2.5 mm in the same medial lateral plane. You repeat the process all over again now looking more closely for optic track, when your moving anterior you are going to be near the optic track, or you would have looked for optic track first time you are probably not going to find it. If you look for it to you get it, you can. Uhm you will go down again to the bottom of GPi, and maybe even a little bit deeper because within the the lesion probe whenever you put it down often times there is this uh uh it does not register well with your micro electrode sometimes. Micro electrode may be at a depth of lets say 75 and you may find optic track at 76. When you go to make a lesion with this you may go down to 76 and stimulate and you may not find optic track until 1.2 to 2 mA it is playing safe at that point. So we'll lesion a little bit deeper than the bottom of GPi again lesioning. While I'll be lesioning at 70 degrees, probably starting at 60 then going up a mm or two doing it again at 75 or 80 and then going up again. Say you got 7 mm which is unusual but you might at 21.5 you are going to get about 6. you probably go

make first lesion then come up about 1 1.5 then middle lesion 4.5, come up 2 mm or 1.5 mm and make another lesion at 70 and that would be it. Then you got a plane lesion, what you do that you take the anterior lesion track and move it medial 2 2.5 mm in occasionally you might go back .5 mm posterior, but usually we just go medial 2 2.5 mm. Then you repeat the process all over again. Remembering in this case however that you now moved to probably about 19.5 and as you go medial in the structure optic track tends to lie as does GPi, if you haven't done a track there you have to assume, which is safe given our past experience that you can, you got to watch your depth you can't go as deep your are gonna find optic track sooner. So you are not going to be as deep on this lesion but your going to go higher. The GPi will come probably 2mm higher. For ever mm you go medial GPi is going to rise about 1 based upon the ap, every mm you go lateral you it is going drop about 1 based upon the ap. And that is the strategy for lesion. Uhm we don't make st lesions right now, most strategy for that right now. Have to talk to the Cubans. Uh for DBS it is a different ballgame. For uh stimulation what you are gonna do is GP's, the problem with stimulation is that you might activate the fibers long before you activate the cells. Or activate the output of the cells so uh you have a problem putting in the DBS same place you put the lesion because you are getting to close the internal capsule. You are gonna activate that long before you get a therapeutic affect um so the voltages are going to be compromised. So what we do our approach is going to be to put it a little bit more lateral and a little bit more anterior within GPi than what you would do a lesion. And so you basically place our lead somewhat on the plane of 21.5 and if it is a Parkinson's patient, if it is a dystonia patient I place it closer to 22 and then we will be anterior by about 4 mm from posterior border of GPi and maybe a little more in uh put the lead down so that in GPi the bottom contact is at the bottom of GPi. Then you do mac stimulation and the mac stimulation is to use to basically confirm

that you got a descent benefit although on GPi the benefit may be delayed. Uh once you have the lead in place you check the mac and you check for side affects. You really checking to see what capsule look like what optic track looks like. Optic track is not as critical, you just don't want to be sitting on it. So if you have a really low threshold at contact 0 of 2 stimulation bipolar at say 200 Hz 90 microseconds then the tend to bring it up a little bit. That has never happened to me. So usually that's ok. Question is how close am I to internal capsule? Am I to posterior am I to medial. The way we find that out is again macro stimulation looking for capsule responses. Again setting it to 2 Hz 90 microseconds, taking your hand on a screen dial flipping from 0 to 2 Volts, 2 to 4 Volts and really just looking just quick up and down up and down, so you can listen to this contraction. If you go slow your going to get a tonic or contraction you cant see. Or if you cant see it it is uncomfortable to the patient. But it is going to harder to see. So you flip 0 to 2, 0 to 2 you don't see anything you go 2 to 4 2 to 4 real quick you don't see anything go 4 to 6 4 to 6 keep doing that till you see something. If you don't see something you are going to have to ask your self why am I not getting anything at 10 Volts? It must be to anterior or to lateral. Then you debate whether you want to move back or medial. Hopefully with the mapping and if there is no lack of registration like between pulling up the micro electrode and sticking down the macro electrode which there is many times based don how this technique is performed right now at both centers. Then what you have to do is look what is my threshold, is it a reasonable threshold. And what you want is that you want to get a therapeutic affect at a voltage that is significantly lower then where you get side affects. Might get a therapeutic affect at 3 Volts and a side affects at 6 Volts that might be ok. But I prefer in GPi not to get macro stim capsule until I get to 7 Volts. Uh that's 7 Volts at 130 Hz if you are at 200 Hz then 7 volts is certainly adequate in most cases. However this may change based upon

review of our own research so. That is what you do for GPi. Lets go over STN again you find the anterior border, you find the lateral border you go in 2.5 3 from the anterior border 2.5 to 3 from the lateral border uh then you place the lead there. Now the strategy for that is that you are trying to affect the sensory motor portion of STN, now whether that is what you are really affecting or whether your affecting … top of STN all of this is still in for debate, we are really not sure clearly this seems to be the best location in our hands, and so we will put it there. And then you have to say well am I really there and you can confirm that with macro stim. Two things you are look for again in STN you are going to see a therapeutic affect there is no delay like there is in GPi and so you see something you are happy with it. If you don't see it you should think where am I uh you need to look for your side affects to see if you get a therapeutic affect. Your happy with that. At 4 Volts you start to see motor and you are not happy with that so you may want to move your lead a little lateral, because ocular motor is medial. You get capsule at 2 Volts you say I am anterior or I am lateral, hopefully your more lateral than anterior because you can it is easier to move things. Uh so I take that back that is not necessarily true I mean, the ap is probably usually move iit back you can see the verbosity. Right now for us they don't allow us to how far medial or lateral that s the problem. But generally speaking when you get a capsule affect you are usually lateral so what we'll move the thing a little more medial. I hope we don't have to move it at all, I hope we get a good therapeutic affect at 2 Volts 3 Volts maybe even 4 Volts bipolar if you do and you don't get a side affect you get to 7 Volts or 8 Volts fine, it is no problem. If I don't get any side affects I don't care as long as I get a good therapeutic affect. That is our strategy for implantation and STN. That's question 1, spent a long time with question 1.

Question 2a what setup procedures, initializations, precautions are you concerned with prior to the first track? Setup procedures uh well one thing in general is with the anesthesia. If the anesthesia sets up on, if you are doing a left procedure for the right body, you like to have all the lines running on the left side. And sometimes based on the anesthesia may not be experienced with your, with how you approach thing they may put your lines on the other side. This is not good because you get in the way of limbs, you don't want to have things limbs in the way. Initializations uh well just setting up the electric physiological equipment, making sure that everything looks good, there is not any problems with the system. Uh precautions, uhm these are basically the things that the patient not given anesthesia, if they are wide awake and they are doing ok, if they are stressed or distressed your system is setup, the map equipment is there, and everybody is ready to go. If you got enough electrode, you should not have anything wrong for path, and you've done your homework, you know what your status is. Some patients targeting is not very well, so you have to be cautious of the patients level of anxiety, and I think lot of us, a lot of people tent not to think much about it because most patients are asleep in the or. These patients are wide awake, and very anxious you can imagine being in there having probes stuck in your brain. So you have to be compassionate, warm, caring and let them know that your are ok, and um and you are moving on, things are going well, keep them informed as what is happening. When you walk away and you go to the back room, which we do to look at our maps, you don't want to start saying "gees where are we." Cause they can hear and understand and um you let the patient know ahead of time that there are going to be some debates of where these tracks are, where your placements are and that this is all perfectly normal and natural and so they are all eased about that. Um if you want to have good anesthesia will attentive some people are some people aren't, most of ours are but in general there

could be some, they kinda sit back don't pay attention to the patient because they are used to the them being asleep. And the patients are asked them for something and uh you hear them and so you take care of it. But the bottom line is they should take care of it so that you can continue with your business. That's 2a.

Question 2b. Is there anything that you check or validate in the operation prior to the track execution? Yes I think it is almost the same as I said in 2a. Just make sure everything, that all the equipment is operating ok, that you got your path set, electrodes are in place. Exterior wires in place, replace preamp wires that may bend or brake, you need extras of everything, you don't have an opportunity to go up and make something if something is broken. So really you need a backup, backup motor, backup of your xy stage, I mean it is nice to have that. Sometimes you can't afford it though. I think its ideal to do. Ok.

Question 3 . this question involves looking through patient records of a track procedure. Given the following 3 pages of an actual patients Pallidotomy. Please describe the first track execution. Limit the discussion from the time that the probe enters the brain, to the point that the probe is retracted. Include explanation and reasoning, for patient interaction (I know we've talked about that), depth observation, cell type classification, layer determination and precautions taken throughout the procedure. Please talk aloud. All right.

Knowledge Engineer 2:

And here are the three pages.

Expert 2:

This is some sort of our tracks. Alright, what I am looking at here is basically a track. And we have time frames, timestamps on here. We have a depth stamp, the depth stamp on this track was basically somewhere arbitrary, we started at depth 30 which does not necessary reflect 30 below the cortex but it is arbitrary number that we started of with that, with our own system. We started of at 30, that's where we started. Then you mark down what you found, the layer basically indicates what's the cell type or are you in Striatum, GPe, GPi. So a type within that area in Striatum you can get things called popcorns, in GPe you get things called pausers and you get things called bursters, in GPi you can get very tonic although we all know it is that straight forward necessarily. First sites, are where you actually collect the cell uh to look at spontaneous activity. Patient interaction is kinda what we are doing at the time. There are comments if we have any interesting observations about that cell in relation to patient symptoms or anything else you want to comment on. So going through this track at depth 50 you found Striatum, it should be continuous down to 56, so that is 6mm. At 57.2 we talked about GPe at 60 we talked about Striatum again which I find highly unusual actual. Cause once you go through Striatum and you go to GPe there is no reason to go back to Striatum, so I am curios about all that. I would find it somewhat unusual to go from 50 to 61 all being Striatum, that is 11 mm unless you where lateral or very anterior cause that's where a lot of the Striatum would be found so it is possibility. Sometimes in Striatum you find these funny little cells that sounds like a burster and you can be confused and call it GPe, it is possible it is what happened here. As you go further down uh again you find GPe at around 62 it looks like. And there is a clear pauser and burster. Once you see the burster you know you are in GPe, except that occasional one in Striatum that could confuse you. So looks to me like that GPe starts around 62 and

goes down you see bursters low frequency discharge pausers, bursters, burster and injuries, lot of pausers and then a border cell, then another pauser, and then a border cell. The border cell starting to indicate that you may be close to getting out of that structure. So I think the call of GPe at 57 was probably not right. I think the call of 62 was because now we are going down to, well maybe not, yeah 67 before we get GPi. That would make sense generally speaking when you are looking for GPe, unless there is some anatomical abnormality, you are going to find about 5mm of GPe. Sometimes 4 sometimes 6. but if you are starting to get 7 or 8 the only possibility there is you are at the very front border and you are at an angle that is coming through a GPe, usually that kind of read or you are lateral. And if you are lateral you are out of GPi and everything is GPe. Now that would be my thinking of that. But in this I can probably dismiss GPe at 57. Cause again we have Pauser right again at 66.4 that is still GPe, although it is not label that way. So it is. You get all these borders, you are getting out and at 5 mm you get borders, your getting out there should be a quiet period. From 67.3 where there was a border there is nothing there till 68 where it says tonic, GPi that was probably a lamina although it was not stated on the track, it is probably what it was. It would be good sometimes write that down and see if it was a lamina, when we have quiet. We mark that down now usually. GPi at 68 uh and basically GPi goes from 68 to 71, so they only found only found about close to 4mm GPi. We have cells that are tonic and pausing in there. Did some motor sensory exams, move well actually no there aren't. This is well moving the arm up, the arm up move arm elbow flexion. This is basically, suppose to be really let the patient. Usually what we say now is that we have a form that clearly marks now that this is motor sensory response elbow flexion, the extension. What that means is that the cell isolated in GPi responded to that particular type of movement. You are looking cause that makes you happy cause you know at essentially a portion of GPi tells

you postal lateral, at least to some degree. Well let's see what else we have. We have a 72.2 a border, alright so quiet 71.9 you think you might be getting out so another border cell suggest you probably are. It is quiet all the way down to 74 so you are probably out of GPi at that point. So then what you want to do is you want to say where am I in GPi, and am I near Capsule, could I be near Optic Track so you shine the light in their face trying see if you can hear the response to a visual path way, the Optic Track. You don't see that then you are going to stimulate and listen to see if you can't see capsule, elicit capsule. If you don't especially if you are done with your track you go back and you plot it out, ok here is track one. So I just looked at track 1 and I see. Track 1 was plotted on the parsadudal planes. Look like a best fit at 21.5 anteriorly, purely because there was no internal capsule. And we did see, yeah we did see optic track stimulation with track 1. So given that you have the track you are probably going to have to go way posterior because you because you get capsule with that and if you didn't get capsule first but the optic track this suggest you are a little more anterior, usually move about 3 posterior. That track is at 21.5 the anterior portion cause there is a small little of GPi that comes up. Subsequent tracks looks like subsequent track move posterior. Found GPe GPi, as we went more posterior found less GPe I assume probably found internal capsule. The hypothesis then is that you are at 21.5 that would give you the best fit. That gives you the best fit, yeah this wasn't really, he made four lesion passes. What you find there, what you want to say ok I think I've got the ap pretty well defined I got at least the front pretty much the back. And it looks like 21.5 then we want to test our hypothesis. Is it 21.5. If it is 21.5 and you move lateral 3mm you are going to 24.5 you get nothing but GPe. That looks like the case here moved 3mm no GPi found at all, a lot of GPe. In this case I think at that point I would now do my mapping because I would move medial a mm to find a small amount of GPi and tell me at 23. uh what we

do then is that we just want to confirm that, make our selves feel better, by getting a longer run in GPi. Then there is another track medial at around 20 , and found a nice long run. I'm sure that made us feel good. Then we decided we can make the lesions. And again using the same strategy as the first lesion passes, right on that first point that is on 21.5 which is what I suggested earlier we probably do. Two lesion passes where made there, at this time we actually made four lesion passes. The next one was made 2.5 mm medial, the first lesion more anterior than the following a little more posterior. Reviewing all our stuff we did not think four lesions made the lesion any bigger than 3 lesions, so we went back to just making just 3 lesions. Ok. That is question 3.

Given the next two pages does this reinforce or change beliefs or findings in question 3? Alright. So this is a lesions track this must be our current that was used stimulation frequency and what we saw when we did the stimulation. So depth is 69 which, can I see those track things again so that I can compare this to. Alright. We were saying that the bottom of GPi was around 72 and here at 69 is our first lesion pass was 0.5 mm behind track 2 alright so we are little bit behind track 2 and track 3 we got down to 69 at 1mA for capsule. Got down 70 70.7 72 is 0.5 ok so you got that at 72. The depth you got was uh 72 so that probably ok. Let's see the depth for track track 2 is probably lesion. Uh possibly could have been a little bit more posterior when you put the lesion there. We may have lesions there, probably did. I think being posterior probably been a small lesion. Bottom line is you basically found could capsule so you know you are posterior that confirms that. It's just to me that the first lesion track at posterior 2.5 it's probably ok I mean that's probably ok. Caus you do spread the capsule remember the capsule is also the spreaded out it grows more as you go medial so that's fine so we obviously made a lesion there. And then made a lesion we got down to 72 because our

impression was .5 you know there is a debate about .5 being ok or not I mean you think you can do it. And actually it seems to me that there is more dyskenesia which is a good sign. (cell phone rings) brining back to 71 you got a 75 no a 60 then 75 degrees really 75, that's 3 4 so at that point we probably got we got enough… uh this is our superimposed lesions upon the track. Yeah I would say based on these that that is right. Ok. You would be ok.

Ok what activities are involved in where the next track should be placed? One rule we made we did after about 10 Pallidotomy's we broke it occasionally the next 400 but I think it is a good rule and that is that your in a plane if you make your track make a run then you find a little bit of each structure uh you found Optic Track, you didn't find capsule you know you are going to have to go posterior and the goal again is to find the posterior lateral border you want to find the posterior border. So you are going to move posterior and how far you move may depend on how much GPi you have. Generally speaking we may move 3mm. GPi is a big structure it breaths easy combinational move unless you have very small one that would tell you capsule your posterior and you would move forward a little bit in that plane. But generally speaking with this kind of track I would move posterior based on what I saw small run GPi got of the track move posterior. Keep going back until you get the internal capsule know what your ap is. And that is what I would have done and that is I would decide that.

What would be your recommendation where the next track should be placed? Well once again I would go posterior 3mm because you have small run of GPi and Optic Track. That is what I would do that.

Anything else you would like to add regarding track execution/procedure? Remember your level of confidence on what you are reporting. It is not always so straight forward that GPe stops here and GPi starts there. Uhm if you get pausers in GPi that can happen. We like to say that there is this type of activity in GPe this type in GPi but the bottom line is there is a grunt population of cells in both structures. Some cells in GPe are faster than you know what and some cells in GPi are not timed and a little slow in the process. And uh what you look for is what the population tells you and it is a population vote. One cell tells you it is slow it does not necessarily mean it in GPe. If the acquired cells are fast and chugging related to sensory motor style it could also be GPe but GPe you tend to find that a little more posterior lateral then you could probably say I am in GPi. But your looking for laminas, your looking for all these things everything goes together, it is not just finding a cell that tells you I am here, it is everything that you found before hand everything put together it is the whole story. That is what's going to help you. There might be some fraction that you are not sure about. When your not sure about it you mark it on the map and say it wasn't clear, I could not tell and you go to the next track. Don't commit yourself to to believing necessarily that you know exactly where your at because you had a track that you thought was pretty clear. It does help, if things are really clear and you have a nice run 5mm GPi and beautiful lamina completely quiet then you get this really high tonic stuff. That's great use these as your landmarks for the next tracks. But there may be cases that are not so clear and you always have little bit of doubt in mind and don't be so confident if your not so sure where you are. Confidence can be a problem sometimes. That is what I would suggest with that.

That was task 2 right.

Knowledge Engineer 2:

Ok let me follow up with a couple of questions on task 2. uh you talked about getting internal capsule responses. What kinda responses are you looking for?

Expert 2:

You are looking for a twitching, muscle contraction. And what you'll see you can see it in the face and tongue. You may see the neck, hand, fingers. Occasionally you'll see the toes, most common place is going to be around the face. But what you want to do is as the patient open their mouth have them relax. Don't let have them stick their tongue out, because when they do that they'll contract their tongue and you may not see some subtle changes. So you just want them to open their mouth, be relaxed, be alert. If they are more sedated the threshold goes up and you may not see it and that is why you also want them awake. The other thing is hold on to their hand while you are doing the stimulations so that you can look in their eyes, look in their face and well look in their mouth look at their face and feel the hand. Then you can tell if the hand is moving the same time because no one els going to be looking at the hand when they are looking at the face. You just hold onto the hand. That is some very subtle things though I mean, I have seen little finger abduction uh I have see a little twitch in the neck muscle. If you don't look for it you don't see it. Uh and it is really important to try and find that it really helps you. I mean there is no other place you can be if you get Capsule other than posterior GPi or you might be medial but still you are posterior in the structure. Because if you go medial GPi goes forward. But it is always the back end that you get the Capsule. So that that is a great landmark to help and as is Optic Track but you are looking ofr that muscle contraction. And again the way to get that is not to hit it down and hold it cause you are going to yank the mouth uh that's obvious, but even if you are

close you are going to get a subtle contraction so you wont see it so well. So clicking it brief pulses just to see the twitch, facial movement you can see make it a lot easier.

Knowledge Engineer 2:

Ok. You talked about I believe was a lack of registration between the micro lead and the macro lead. Can you say what that is caused by, how much are we talking?

Expert 2:

Yeah I think the debate is that there is always a little bit. Nothing is these systems, current systems don't as good as they are they still can improved. Everything is not necisarily real tight. There can be little bit of movement in the system like the xy stage uh there can be a little bit of change in terms of the guide tubes in terms of how big they are. Your electrode can move within it. Electrodes aren't always perfectly straight in the plateranium micro electrodes and they are edged in glass and so they have a little curvature when they go down in the brain they are going to follow that. So the electrode that is fine but when you go to a DBS electrode that's great and it may be accurate. So there may be some lack of registration with one versus the other. You are also taking one system off brining the other system down and a maxi frames don't have fanthoms so you can't always look to see whether they are going to the same place. Uh CRW do so you can so that helps you uh so just because there are some mechanically problems that you can could put your DBS down the same track and the same guide tube that you had your micro electrode down now that would be a plus. Now where all used to do that, and working on that we realize this is a problem with the system. What's more there is nothing to me more frustrating than doing a map, knowing this is a great map, I've got the edges you got it done. You put your DBS probe down and you get some side affect

and you say where'd that come from, and we've all done that. And I mean they tell you, if someone else tells you they have not done that, it means they've done one case or 2 cases but if you do enough cases you'll see it, everybody will no matter how good they are. Uh and it is a problem with our approach we need to make it better and we can with help, like you guys. Oh last thing, brain shift that could happen and may not be related to your mechanics at all. CSV leakage could cause the problem, that is something your gonna have, that is a problem that you are just going to have. Uh you want to minimize that, keep things covered, a lot of saline, things like that.

Knowledge Engineer 2:

Anyway that you can detect whether it shifted?

Expert 2:

Well that is something to think about. Could you put a marker somewhere, sometime you can tell when the Cortex drops down a little bit if CSV if leakage that you may suggest things are shifting a little bit. If there is shift usually the patient is laying flat it is going shift things the brain is going to come backwards and things move in a little bit. So it is going to move you forward and lateral most of the time that is something you want to keep in mind.

Knowledge Engineer 2:

Ok then the last question you had talked about uh regarding question 2 looking at the apparatus and the system prior to the track. What problems do you typically experience

that have to be corrected inside the or, you talked about having to replace the parts, but how do you know one of the parts go bad, or what do you typically happens?

Expert 2:

Well there are two different parts. There is the mechanical things that you want to make sure they are straight. Uh thin guide tubes are unacceptable and you can get that if you are pulling things off, putting things down, laying things down on top of things and not that they need to but that happens it is the or and sometimes the staff is not fully. You may different staffs you know, it is just the staffs are shifts and people do this stuff all the time, and some people don't do it very much. Ideally you'd like to have your same staff all the time to work with the equipment so that you avoid those kinds of problems to. Keep it straight neat and clean that all you want. The other thing is the mechanical brake down of triads. The way that you have to trouble shoot is, there is no no substitute for experience and problem having people to do that is Klaus. Uh helps designs systems, he knows how they work, he knows what the problems can be, he is a pretty overall guy someone like that needs to lay out all the issues about what you look for and what else. But if you are also inexperienced like the physiologist then your also having trouble shooting all these problems. And so if you have noise, think about things that are turned on, and shouldn't be turned on, uh coteries uh sometimes just uh blankest, warming blankets, all kinds of other things. And if you need to turn them off make sure you unplug them. So there are those issues. Uh the preamp wire in certain systems are very soft and friable and can brake after it is bent a few times, you want to keep that wire nice and straight, but obviously people pick things up and hand them to you so that happens after a while, so keep that in mind. It grows me not to be good if you stare at things, sometimes things, there are oxidation that occurs and you don't get could contact

there, you bring a wire and a brush to clear them out. You can have saline at times and it drives people crazy, particularly because it also causes a problem that is a quick fix for us. Buy new wires, wires can go bad. You grab wires and they pull off so connections get loose , the way you did it was to graben the tip of the wire you'd be fine, but nobody does, they just grab the wire and pull it off. So you look at your wires, and just replace things one by one if you have to until you can reduce your noise level. Sometimes it is your microelectrode, sometimes the patient is just moving so much that there is a lot of micro phonics a mechanical joggling things in that cause you have to decide if you have to give your patient something to quiet them down. Keeping in mind that whatever you get from quieting them down may change your neuronal activity, you may be out of luck that way. So those are all some issues.

Knowledge Engineer 2:

Ok anything else that you'd like to add that we may have missed.

Expert 2:

I don't think so. There are probably a lot of stuff that you know I did not think about to say but by and large that a lot of stuff for you learn from experience. You know more every time you do it you find something new half the time so there is no end to the problems that could occur. A lot of moving parts essentially…

APPENDIX C

RULE EXTRACTS

Probable Rules from TASK 2 and TASK 3

As determined from Knowledge Acquisition of Experts

Depth:

- First Layer encountered => Striatum
- If Striatum not encountered => problem with electrode or recording equipment.
- Quiet areas in signal at beginning of track => Non continuous Striatum
- At 60-63 => encounter GPe, At 50 to 62 => usually have ST.
- Lamina thickness, < 0.5mm unacceptable => could have missed the start of lamina.
- GPe thickness is ~5mm.
- GPi usually starts at 70, if GPe starts at 63.
- 1-1.5 mm below GPi ventral border => check for OT/IC
- Depth 30 is arbitrary standard to start at.
- ST typically 6mm thick.
- GPe 4-6mm, mostly 5mm => good targeting
- Lamina => 0.5mm – 1mm in depth
- GPe starts at about the same depth every time.

Cell:

- Not 4 continuous cells in a row in ST => electrode need to be readjusted.
- Bursting & Pausing & tonic & chugging => GPi.
- Burster & Pauser & Border => GPe

- Pausing & Pauser => hard to distinguish
- Bursting & Burster => very different sound
- Chugging => is it associated with tremor? => Yes, then definitely in GPi and lesion there.
- Quiet stretches => Lamina
- Popcorns and caudate Putamen => ST
- Burster in ST =? Possible but not GPe.
- See a clear Burster => Definitely in GPe.
- Border cell => close to exiting structure, cell on edges of structures.
- Pausers in GPi => unlikely but possible
- Population cell type => determines layer.
- Noise interference => check equipment before proceeding. Low background noise => Lamina
- Doctors have to provide input for cell types
- Pauser in GPe and Pausing in GPi => Difficult to discern pattern.
- If signal is lost and returns before 1mm => Continue on
- If signal is lost and does not return after 1mm => Check impedance of the electrode.
- Pausers and tonic hardest to discern.
- Lamina => fibers.

Process:

1. Identify the target
2. Identify where you are within the target
3. Identify the borders of the target.
    - Dorsal, ventral border
    - Posterior lateral border
4. Identify nearby critical structure that you want to avoid.
    - Optic Track
    - Internal Capsule

- Anatomical mapping and sensory motor mapping => probe location along the track.
- Brain shifted => start over with tracking.
- Determine if you are in ST, GPe, GPi.
- If patient has seizure => stop the procedure.
- If patient recovers from seizure => start over with mapping because of brain shift.

PI/ Sensory Motor:

- Sensory motor exploration => no response => note and move on.
- Types of sensory motor exploration
    - Active or passive movement
    - Flexion and extension
    - Wrist, arm, fingers
- Conflicting evidence for OT/IC => note and move on.
- Motor sensory response to elbow flexion => track postal lateral.

- Out of GPi => Test for OT => Yes, then anterior. No then Test for IC => Yes then posterior, No then anterior.
- Twitching, muscle contraction in face, tongue, neck, hand, fingers => IC found.
- If in GPi => Do sensory motor exploration.

  Reasong for sensory motor exploration
  - Determine what sensory section probe is in.
  - Determine whether lesion/DBS will have predictable outcome.
- If in Striatum => no sensory motor exploration.
- If in GPe => sometimes do sensory motor exploration out of curiosity. No benefit to track planning or reasoning.
- Passive movement of extremeties => positive then in sensory motor area.
- Active movement both ipsy and controlateral side.
- Probe back for trunk area reaction.
- Patient tremor relation => GPi.
- OT micro stimulation => positive or negative note for track planning.
- IC => positive or negative note for track planning.
- Stimulating for OT => look for a corresponding activity in signal.

Layer Classification:

- Cannot have ST, then GPe, then ST.
- Quiet stretches => Lamina
- Large background activity => Probe in GPi
- See a clear Burster => Definitely in GPe.
- If GPi not found => Probe can be to anterior or lateral, check electrode, check for hemorrhaging.

Track planning, placement, validation

- 5-8mm ST, high up => medial
- a lot of ST, deeper => lateral
- long run ST => anterior
- short run ST => posterior
- Short GPi run (4mm) => track lateral or posterior edge or anterior edge.
- Long GPi run (8mm) => center of GPi.
- 11mm ST => track very lateral or anterior.
- 7-8mm GPe => very anterior and could miss GPi.
- AP defined => move lateral or medial.
- Short GPi run => to anterior, or posterior
- Found Capsule => at Posterior edge.
- Long run ST and short run GPi => anterior or lateral.
- Move track => Is AP defined => No, and then stay in parsadual plane. Yes, and then step medial or lateral.
- If anterior => move posterior
- Length of GPi determines ap,ml
- If anterior => move posterior
- If lateral => move medial
- If lateral => move anterior or posterior.
- Find OT no IC => move posterior (RULE!!!!!)
- Small GPi and OT => move posterior.
- Keep going posterior until hit IC => found posterior edge.
- Step max 4mm, step min 1mm, step avg 2mm.

- Nice long run through GPi (5mm) posterior means we are medial.
- Uncertainties in track placement for first track.
    - Resolution of MRI
    - Definition of MRI structures
    - Users interpretation of MRI
    - Targeting apparatus can vary in a couple of millimeters.
    - Reading errors of coordinates from STEALTH to stereotactic head frame.

  => Results in a composite error that is always present. User learns to compensate for the error.
- For the next track stay in parasadual plane. Do not make 2 changes at once.

  Note: this might change because of the new 3D modeling tool.
- Each track is unique and can not be referenced to the previous track.

Pre-op / Precautions:

- Craniotomy open to long => increase risk of seizure or hemorrhaging.
- Electrode is resonating => interference with signal => check equipment.
- Backup of xy stage
- Path plan is set before track.
- No GPi found => check approach, for micro bleeds, or equipment.
- Found Capsule => Found posterior edge

APPENDIX D

KBS CODE

D.1. Loading File

```
;---------------------------------------------------------------------------------------------------
;                       Loading File
;       Program:        Pallidotomy Version 6.0
;       Author:         Linda Harley
;       Date:           November 23, 2003
;       File:           pal.bat
;---------------------------------------------------------------------------------------------------
(load "template.clp")
(load "facts.clp")
(load "interface.clp")
(load "topview.clp")
(load "preop.clp")
(load "initial.clp")
(load "onetrack.clp")
(load "eventwakeup.clp")
(load "depth.clp")
(load "dsp.clp")
(load "cellanat.clp")
(load "user.clp")
(load "comparecell.clp")
(load "inferlayer.clp")
```

```
(load "trackplanning.clp")
```

D.2. Template

```
;       ---------------------------------------------------------------------------------------------------
;                       Template
;       Program:        Pallidotomy Version 6.0
;       Author:         Linda Harley
;       Date:           September 1, 2003
;       File:           template.clp
;       Updates:        Sep 1, 2003     Starting reconstruction of KBS.
;                       Sep 4, 2003     Created template "new_pathplan"
;                       Sep 11, 2003    Created templates "cell_layer_combo"
;                                       "deciding_cellid_layer", "dsp_max", "dsp_lsit".
;                       Oct 7, 2003     Added the "wakeup_depth" template.
;                       Oct 24, 2003    Added the "layer_thick" and "apml_assign".
;       ---------------------------------------------------------------------------------------------------

;GLOBALS
;---------------------------------------------------------------------------------------------------------
;These are specified to handle the output changes from the CLIPS terminal to the
;C++ handler through ACTIVE X control.

;(defglobal ?*output_device* = t)       ;use when running KBS in CLIPS.
 (defglobal ?*output_device* = wdialog)     ;use when running KBS with Onetrack
                                            ;software.
```

```
;TEMPLATES
;---------------------------------------------------------------------------------------------------------
;This is the format in which the C++ will be asserting information into the KBS.
;Some of these fields are only used at very specific instances, others are used
;throughout the KBS program.  Note:  If multislots are not being used they need to have
;'nil' in their slots when you are doing an assert command.

(deftemplate answer
        (slot yn-answer)        ;Generally the KBS asks a yes/no question and wants an
                                ;answer.
        (slot multichoice)      ;The KBS asks a multiple choice question and wants an
                                ;answer.
        (slot timetag)          ;Assigned by C++ to make sure we are always getting new
                                ;info.
        (slot current-depth)    ;Current depth value of the probe.
        (multislot text-field
                (default nil) ) ;The user is allowed to enter text.
        (multislot pathplan
                (default nil) ) ;Always provided at the beginning of the track.
        (multislot dsp-cell
                (default nil) ) ;Cell type probabilities list as determined by DSP analysis.
        (slot format)           ;Used when answering a specific request.
        (slot status)           ;Used for varifying that message was received.
        (multislot user
                (default nil) ) ;For direct input of the user to the KBS.  User initiated.
```

```
)
;For the time being this will be the same format.  Will change as we start implementing
;into the actual program.  This template serves to place answers in correct format after
;receiving input from C++.

(deftemplate answer_received
        (slot yn-answer)        ;Generally the KBS asks a yes/no question and wants an
                                ;answer.
        (slot multichoice)      ;The KBS asks a multiple choice question and wants an
                                ;answer.
        (slot timetag)          ;Assigned by C++ to make sure we are always getting new
                                ;info.
        (slot current-depth)    ;Current depth value of the probe.
        (multislot text-field
                (default nil) ) ;The user is allowed to enter text.
        (multislot pathplan
                (default nil) ) ;Always provided at the beginning of the track.
        (multislot dsp-cell
                (default nil) ) ;Cell type probabilities list as determined by DSP analysis.
        (slot format)           ;Used when answering a specific request.
        (slot status)           ;Used for verifying that message was received.
        (multislot user
                (default nil) ) ;For direct input of the user to the KBS.  User initiated.
)
```

```
;This next template will be used primarily in the handling between the general rules
;and the output rules set, that handles communication out of KBS to the C++.
;This template is still in fluctuation and will change as needed.

(deftemplate output
        (slot type)             ;this will specify what kind of action we are expecting.
        (slot format)           ;the format that we are expecting to receive the answer.
        (multislot variable
                (default nil) ) ;the variables that we want to display for the user to select.
        (multislot text-field
                (default nil) ) ;if a question accompanies output, this is to be displayed.
        (slot depth)            ;when submitting data will need to know what depth we are
                                ;referring to.
        (slot timetag)          ;if we are referring to data that came in, C++ needs to be
                                ;able to correlate.
)

;Some initial information of the patient needs to be gathered.  This info will be
;stored with the use of the following template

(deftemplate initial_info
        (slot name)             ;the specific field such as doctors, name or date...
        (slot value)            ;the info that refers to the above field.
)
```

```
;The anatomy_layer_down and anatomy_layer_up is used to set up the format in which
;the facts regarding combinations of layers will be set up as, during deffacts.
;These templates will always stay like this, because the knowledge have not changed.

(deftemplate anatomy_layer_down
        (slot previouse)
        (slot down)
)
(deftemplate anatomy_layer_up
        (slot previouse)
        (slot up)
)

;The following template is used to store the pathplan of a particular track in. This will
;keep it generalized for now assuming that we only get the pathplan info at the beginning
;of a given track.  May have to include a slot when we want to update the pathplan
;during the execution of a track.  NOTE:  this template will most likely change when we
;start to take into consideration the uncertainties related with the path plan.  For right
;now we are assuming that the pathplan is 100% accurate and fixed.
(deftemplate pathplan
        (slot name)             ;the name of the layer
        (slot start)            ;the starting z depth of the layer
        (slot end)              ;the ending z depth of the layer
        (slot track_id)         ;the track that this path plan is specified for.
)
```

```
;The following 3 rules are added to make the current program run.  These will change as
;we start to change their reasoning when we come to it.  The CONTROL template is
;used to make sure that we are able to link each new entry with the previouse and next
;entry.  This template is closely associated with "Event Output" and will be used to help
;produce the "actual path plan."
(deftemplate control
        (slot name)
        (slot location_id)
        (slot previouse_id)
        (slot track_id)
        (slot depth)
        (slot time)
        (slot layer)
        (slot cell)
        (slot state)
)

;The EXPECTING LAYER is  used in reasoning with just the current depth value.
(deftemplate expecting_layer
        (slot name)
        (slot location_id)
        (slot previouse_id)
        (slot timetag)
)
```

```
;The DECIDING LAYER is the place holder for each sub phase decision of what layer
;the probe is in.  Template has changed to incorporate new process layout.
(deftemplate deciding_layer
        (slot depth)            ;based purely on depth pathplan reasoning
                                ;for now only one used.
        (slot cell)             ;as determined by cell
        (slot user)             ;as determined by user
        (slot decision)         ;final decision on what layer the probe is.
        (slot location_id)      ;unique id
        (slot timetag)
)

;NEW PATHPLAN is used for updating the new pathplan for a speacific track as the
;track progresses.
(deftemplate new_pathplan
        (slot ST)
        (slot STstate)
        (slot Le)
        (slot Lestate)
        (slot GPe)
        (slot GPestate)
        (slot Li)
        (slot Listate)
        (slot GPi)
        (slot GPistate)
```

```
        (slot La)
        (slot Lastate)
        (slot OT)
        (slot OTstate)
        (slot IC)
        (slot ICstate)
        (slot track_id)
)

;The CELL LAYER COMBO contains the combinations between cell type layers and cell
;types.  This template will never change because the knowledge will does not change.
(deftemplate cell_layer_combo
        (slot layer)            ;layer such as ST, GPe,...
        (slot cell)             ;the cell type associated with the layer
)

;The DECIDING CELLID LAYER is the place holder for the different phases to place
;there decision on what cell type is being observed.  From this we can then infer what
;cell type layer is expected.
(deftemplate deciding_cellid_layer
        (slot anat)             ;based on anatomical known truths
        (slot dsp)              ;as determined by DSP
        (slot user)             ;as determined by user
        (slot prevhist)         ;based on previous decisions
        (slot choice)           ;final cell type chosen
```

```
        (slot override)          ;if the user decides to over ride
        (slot location_id)
        (slot timetag)
)

;The DSP MAX collects the top 3 choices from the DSP probability list.
(deftemplate dsp_max
        (slot max)
        (slot 2max)
        (slot 3max)
        (slot status)
        (slot location_id)
        (slot timetag)
)

;The DSP LIST serves as the template in which the probability array of the cell types are
;specified.
(deftemplate dsp_list
        (slot sn_ratio)
        (slot background)
        (slot caudate)
        (slot popcorn)
        (slot HFD-P)
        (slot LFD-B)
        (slot border)
```

```
        (slot tonic)
        (slot bursting)
        (slot chugging)
        (slot pausing)
        (slot fiber)
        (slot multiple)
        (slot artificial)
        (slot injury)
        (slot tremor)
        (slot neg)
        (slot location_id)
        (slot timetag)
)

;The USER is the template in which the user input at the beginning of an event is stored
;in.
(deftemplate user
        (slot layer)
        (slot cell)
        (slot location_id)
        (slot timetag)
)

;The WAKEUP DEPTH template is used to regulate the depths at which the KBS needs
;to be awaken, for verification purposes.
```

```
(deftemplate wakeup_depth
        (slot ST_start)
        (slot ST_half)
        (slot ST_nend)          ;nend indicates near end
        (slot Le_half)
        (slot GPe_start)
        (slot GPe_half)
        (slot GPe_nend)
        (slot Li_half)
        (slot GPi_start)
        (slot GPi_half)
        (slot GPi_nend)
        (slot La_start)
        (slot OT_start)
        (slot IC_start)
        (slot track_id)
        (slot last_limit);the last limit set.
        (slot status)           ;to keep track of what has been done.
)

;The LAYER THICK template is used to store the primary layer thicknesses calculated
;from the new_pathplan.
(deftemplate layer_thick
        (slot ST)
        (slot GPe)
```

```
        (slot GPi)
        (slot track_id)
)

;The APML ASSIGN template is used to assign the respective apml associated with
;each of the primary layers and OT and IC.
(deftemplate apml_assign
        (slot ST)
        (slot GPe)
        (slot GPi)
        (slot OT)
        (slot IC)
        (slot status)
        (slot track_id)
)
```

D.3. Facts

```
;       ---------------------------------------------------------------------------------------------------
;                       Facts
;       Program:        Pallidotomy Version 5.0
;       Author:         Linda Harley
;       Date:           September 2, 2003
;       File:           facts.clp
;       Updates:        Sept 2, 2003    Starting Reconstruction of KBS.
;                       Sep 11, 2003    Added known_cell_layer_combinations
;                       Sep 13, 2003    for cell_layer combo made cell border
;                                       possible at any layer.
;       ---------------------------------------------------------------------------------------------------
;FACTS
;---------------------------------------------------------------------------------------------------------
;Facts are important for they set up known specifics in the knowledge.  They represent
;known facts about the biology of the brain that can not change.  They are used primarily
;in rules that will not fluctuate, for the knowledge is known for certain, how it is used and
;inferred upon my vary.

;ANATOMY
;---------------------------------------------------------------------------------------------------------
;The order in which the structures of the brain appear in as the probe progresses
;through the brain is known.  "downward_layering" refers to the order in which layers will
;be encountered as the probe moves deeper into the brain.  "upward_layering" refers to
;the order in which layers will be encountered as the probe moves out of the brain.
```

```
;These are all known combinations.  Not all of them might be in use at the moment, but
;is still possible.

(deffacts downward_layering
        (anatomy_layer_down (previouse none) (down ST) )
        (anatomy_layer_down (previouse none) (down GPe) )
        (anatomy_layer_down (previouse none) (down Le) )
        (anatomy_layer_down (previouse none) (down Li) )
        (anatomy_layer_down (previouse none) (down GPi) )
        (anatomy_layer_down (previouse none) (down La) )
        (anatomy_layer_down (previouse ST) (down ST) )
        (anatomy_layer_down (previouse ST) (down Le) )
        (anatomy_layer_down (previouse ST) (down NB) )
        (anatomy_layer_down (previouse ST) (down GPe) )
        (anatomy_layer_down (previouse Le) (down Le) )
        (anatomy_layer_down (previouse Le) (down GPe) )
        (anatomy_layer_down (previouse GPe) (down GPe) )
        (anatomy_layer_down (previouse GPe) (down Li) )
        (anatomy_layer_down (previouse GPe) (down GPi) )
        (anatomy_layer_down (previouse Li) (down Li) )
        (anatomy_layer_down (previouse Li) (down GPi) )
        (anatomy_layer_down (previouse GPi) (down GPi) )
        (anatomy_layer_down (previouse GPi) (down La) )
        (anatomy_layer_down (previouse GPi) (down OT) )
        (anatomy_layer_down (previouse GPi) (down IC) )
```

```
        (anatomy_layer_down (previouse La) (down La) )
        (anatomy_layer_down (previouse La) (down OT) )
        (anatomy_layer_down (previouse La) (down IC) )
        (anatomy_layer_down (previouse La) (down NB) )
        (anatomy_layer_down (previouse OT) (down OT))
        (anatomy_layer_down (previouse OT) (down IC))
        (anatomy_layer_down (previouse IC) (down IC))
)
(deffacts upward_layering
        (anatomy_layer_up (previouse ST) (up ST) )
        (anatomy_layer_up (previouse ST) (up none) )
        (anatomy_layer_up (previouse Le) (up Le) )
        (anatomy_layer_up (previouse Le) (up ST) )
        (anatomy_layer_up (previouse GPe) (up GPe) )
        (anatomy_layer_up (previouse GPe) (up ST) )
        (anatomy_layer_up (previouse GPe) (up none) )
        (anatomy_layer_up (previouse GPe) (up Le) )
        (anatomy_layer_up (previouse Li) (up Li) )
        (anatomy_layer_up (previouse Li) (up GPe) )
        (anatomy_layer_up (previouse GPi) (up GPi) )
        (anatomy_layer_up (previouse GPi) (up GPe) )
        (anatomy_layer_up (previouse GPi) (up Li) )
        (anatomy_layer_up (previouse La) (up La) )
        (anatomy_layer_up (previouse La) (up GPi) )
        (anatomy_layer_up (previouse OT) (up OT) )
```

```
        (anatomy_layer_up (previouse OT) (up La) )
        (anatomy_layer_up (previouse OT) (up GPi) )
        (anatomy_layer_up (previouse IC) (up IC) )
        (anatomy_layer_up (previouse IC) (up La) )
        (anatomy_layer_up (previouse IC) (up GPi) )
        (anatomy_layer_up (previouse IC) (up OT) )
        (anatomy_layer_up (previouse NB) (up NB) )
        (anatomy_layer_up (previouse NB) (up La) )
        (anatomy_layer_up (previouse NB) (up ST) )
)

;CELL TYPES
;-------------------------------------------------------------------------------------------------------------
;Through experiments over the years it has been determined what type of cell to expect
;at each layer in the brain.  The combination between cell type specification and layer
;dependency is thus represented within fact "known_cell_layer_combinations".  Once
;again these are known stable truths, the wording might change and there could also be
;some additional ones added later.
(deffacts known_cell_layer_combinations
        (cell_layer_combo (layer ST )(cell caudate))
        (cell_layer_combo (layer ST )(cell popcorn ))
        (cell_layer_combo (layer ST)(cell border))
        (cell_layer_combo (layer Le )(cell fiber ))
        (cell_layer_combo (layer Le )(cell border))
        (cell_layer_combo (layer GPe )(cell HFD-P ))
```

```
(cell_layer_combo (layer GPe )(cell LFD-B ))
(cell_layer_combo (layer GPe )(cell border))
(cell_layer_combo (layer Li )(cell fiber ))
(cell_layer_combo (layer Li )(cell border ))
(cell_layer_combo (layer GPi )(cell tonic ))
(cell_layer_combo (layer GPi )(cell bursting ))
(cell_layer_combo (layer GPi )(cell chugging ))
(cell_layer_combo (layer GPi )(cell pausing ))
(cell_layer_combo (layer GPi )(cell tremor ))
(cell_layer_combo (layer GPi )(cell border))
(cell_layer_combo (layer La )(cell fiber ))
(cell_layer_combo (layer La)(cell border))
(cell_layer_combo (layer IC )(cell multiple ))
(cell_layer_combo (layer any )(cell artificial ))
(cell_layer_combo (layer IC )(cell injury ))
(cell_layer_combo (layer IC )(cell neg))
(cell_layer_combo (layer OT )(cell none))
(cell_layer_combo (layer none )(cell none ))
)
```

D.4. Interface

```
;       ---------------------------------------------------------------------------------------------------
;                       Interface
;       Program:        Pallidotomy Version 6.0
;       Author:         Linda Harley
;       Date:           September 1, 2003
;       File:           interface.clp
;       Changes:        Sep 1           Starting reconstruction on KBS.
;                       Oct 7           Added the rule "set_wakeup_out"
;       ---------------------------------------------------------------------------------------------------
;INTERFACE
;---------------------------------------------------------------------------------------------------------
;Common Output Rule:  This rule puts all output in the correct format and filters it
;through to the C++ via the ACTIVE X Control.  NB: This is the only rule that will put the
;KBS to "sleep".  All output throughout the program will be directed to this rule.
 (defrule interface_KBS_out
        ?out <- (output (type ?t)(format ?f)(variable $?v)(text-field $?tf)(depth ?d)
                (timetag ?time))
        ?goal <- (goal interface_initiated status active)
=>
        (retract ?out)
        (retract ?goal)
        (assert (goal interface_initiated status complete))
        (printout wdialog ?t":"?f"*"?v"*"?tf"!" crlf)
        (halt)  )                       ;This halt is extremely important.  For it signals
```

```
                                        ;that the KBS is waiting for some form of input.
                                        ;This is one rule that puts the KBS to "sleep".
;This unique output rule only fires when assigning the wakeup depth.  Important to note
;that it does not halt the program but only puts the information in the buffer. The C++
;program only reads the buffer when the KBS is halted, thus multiple strings can be
;there.
(defrule set_wakeup_out_Data
        ?out <- (output (type ?t)(format ?f)(variable $?v)(text-field $?tf)(depth ?d)
                        (timetag ?time))
        ?goal <- (goal set_wakeup status complete)
=>
        (retract ?out ?goal)
        (printout wdialog ?t":"?f"*"?v"*"?tf"!"crlf)
        (assert (event_subgoal name set_wakeup status complete))
)

;Common Input Rule:  This rule puts all the input from the C++ in a usable format for the
;KBS to use in reasoning.  With implementation might divide this rule up into different
;categories i.e. whether the input was KBS initiated or not.  This rule should definitely be
;divided into at least two subsets. (1) that handles information coming in when the KBS
;asked for information of a particular nature. (2) when a new site is identified, the user
;input becomes available, and/or a certain depth was reached and KBS needs to be
;awakened for a decision, and initial pathplan was decided. This second one we will call
;"KBS_wakeup".
```

```
(defrule interface_KBS_in
        ?goal <- (goal interface_initiated status complete)
        ?answer <- (answer (yn-answer ?yn)(multichoice ?mc)(timetag ?time)
                                (current-depth ?d)(text-field $?text)(pathplan $?path)
                                (dsp-cell $?cell)(format ?f)(status ?stat) )
        ?avail <- (goal answer status available)
=>
        (retract ?goal)
        (retract ?answer)
        (retract ?avail)
        (assert (answer_received (yn-answer ?yn)(multichoice ?mc)(timetag ?time)
                                        (current-depth ?d)(text-field ?text)(pathplan ?path)
                                        (dsp-cell ?cell)(format ?f)(status ?stat)))
)

;Common Wakeup Rule:  This rule is specially created for when the system wants to
;wake up the KBS.  KBS is "woken up" when an event occurs.
(defrule KBS_wakeup_in
        (goal name onetrack status active)
        ?wake <- (goal name kbs status wakeup)
        ?answer <- (answer (timetag ?time)(current-depth ?d)(dsp-cell $?cell)
                                (user $?user)(format ?f)(status ?stat))
=>
        (retract ?wake)
        (retract ?answer)
```

```
        (assert (answer_received (timetag ?time)(current-depth ?d)(dsp-cell $?cell)
                                        (user $?user)(format ?f)(status ?stat)))
        (assert (subgoal name event_wakeup status active))
        (assert (deciding_layer (depth nil)(cell nil)(user nil)(decision nil)
                                    (location_id nil)(timetag nil)))
)

;END TRACK
;-------------------------------------------------------------------------------------------------------------
;Whenever the user decides to end the track, it will be sent over in the status of the
;answer string.
(defrule end_track_input
        ?avail <- (goal answer status available)
        ?answer <- (answer (status endtrack))
=>
        (retract ?avail)
        (retract ?answer)
        (assert (end track) )
)
```

```
;END OPERATION
;-------------------------------------------------------------------------------------------------------------
;Whenever the user decides to end the operation, it will be sent over in the status of
;the answer string.
(defrule end_operation_input
        ?avail <- (goal answer status available)
        ?answer <- (answer (status endop) )
=>
        (retract ?avail)
        (retract ?answer)
        (assert (end operation) )
)
```

D.5. Top View

```
;       ---------------------------------------------------------------------------------------------------
;                       Topview
;       Program:        Pallidotomy Version 6.0
;       Author:         Linda Harley and Nelson Baker
;       Date:           September 1, 2003
;       File:           topview.clp
;       Notes:          Most of the code is taken from Version 4.0 with some
;                       additions and changes as noted below.
;       Updates:        Sep 1           Starting reconstruction of KBS.
;                       Sep 4           Added cleanup to trackplanning and defrule
;                                       new_track which serves as a mechanism
;                                       for testing in KBS.
;       ---------------------------------------------------------------------------------------------------
;THE INTIAL CONTROL PRODUCTION
;---------------------------------------------------------------------------------------------------------
;This portion of the code is like the driver of a car.  It determines where we are heading
;and the order in which the phases will take place.  The program will start by gathering
;some vital information about the patient during the "pre_op" phase.  During "initialize"
;phase the pathplan is obtained.  Most of the track execution happens during "onetrack"
;phase when we are actually going through the track.  The "track planning" phase is
;there to suggest where next track should be placed and clean up of code.
(defrule setup
        (initial-fact)
=>
```

```
        (assert (sequence current_node pre_op next_node initialize) )
        (assert (sequence current_node initialize next_node onetrack) )
        (assert (sequence current_node onetrack next_node trackplanning) )
        (assert (goal name pre_op status active) )
)

;SWITCH RULES BETWEEN GOALS
;-------------------------------------------------------------------------------------------------------------
;This rule is the mechanism that enables the KBS to switch between phases. When one
;phase is complete the next one will be activated immediately.
(defrule switch_rule
        (goal name ?any status complete)
        (sequence current_node ?any next_node ?next)
=>
        (assert (goal name ?next status active) )
)

;DOING THE GOALS
;-------------------------------------------------------------------------------------------------------------
;The following set of rules set up the LH-side which controls when a phase has been
;completed.  These vary depending on the phase.
(defrule done_goal_pre_op
        ?goal <- (goal name pre_op status active)
        ?hello <- (pre_op_subgoal name hello status complete)
        ?info <- (pre_op_subgoal name patient_info status complete)
```

```
=>
        (retract ?goal)
        (retract ?hello)
        (retract ?info)
        (assert (goal name pre_op status complete) )
)
(defrule done_goal_initialize
        ?goal <- (goal name initialize status active)
        ?got_pathplan <- (initial_subgoal name pathplan status complete)
        ?templates <- (initial_subgoal name templates status complete)
        ?wakeup <- (initial_subgoal name calculate_wakeup status complete)
=>
        (retract ?goal ?wakeup ?got_pathplan ?templates)
        (assert (goal name initialize status complete) )
        (assert (first time) )    ;The track can only be initialized at the actual
                                  ;beginning of the track.
)
(defrule done_goal_onetrack
        ?goal <- (goal name onetrack status active)
        ?donetrack <- (done onetrack)
=>
        (retract ?goal)
        (retract ?donetrack)
        (assert (goal name onetrack status complete) )
)
```

```
(defrule done_goal_trackplanning
        ?goal <- (goal name trackplanning status active)
        ?clean <- (trackgoal name cleanup_track status complete)
        ?layer <- (tpgoal name layerthick status complete)
        ?apml <- (tpgoal name apml_assign status complete)
        ?plan <- (tpgoal name plan_track status complete)
        ?tpclean <- (tpgoal name cleanup status complete)
=>
        (retract ?goal ?clean ?layer ?apml ?plan ?tpclean)
        (assert (goal interface_initiated status complete))
                ;necessary because we are waiting to hear from
                ;C++ what to do next.  a new track or
                ;end the operation.
)
(defrule new_track
        ?goal <- (new track)
=>
        (retract ?goal)
        (assert (goal name initialize status active))
)
(defrule done_goal_endop
        ?goal <- (end operation)
=>
        (retract ?goal)
        (assert (goal name endop status complete) )
```

```
        (save-facts kbsdata.txt)
)

;-------------------------------------------------------------------------------------------------------------
;       add the object needed to start execution

(deffacts initial-data " " )
```

D.6. Pre-Operation

```
;       ---------------------------------------------------------------------------------------------------
;                       Pre Operation
;       Program:        Pallidotomy Version 6.0
;       Author:         Linda Harley
;       Date:           September 1, 2003
;       File:           preop.clp
;       Updates:        Sep 1           reconstruction of KBS.
;       ---------------------------------------------------------------------------------------------------
;HELLO
;---------------------------------------------------------------------------------------------------------
;The following rule will simply display a message in the "message box" on the screen.
;The purpose is to notify the user that the KBS has been activated and is starting to
;process.  Any time a rule name is followed by "_out" it indicates that the rule is
;triggering the interface rule.  Any time a rule name is followed by "_in" it means that an
;answer has been received and will now be modified to be used in the code.
(defrule hello_out
        (goal name pre_op status active)
=>
        (assert (output (type M)(format message)(variable "")
                (text-field "The KBS has been activated and is standing by.")))
        (assert (goal interface_initiated status active))
)
```

```
(defrule hello_in
        (goal name pre_op status active)
        ?hello <- (answer_received (timetag ?time)(format message)(status received))
=>
        (retract ?hello)
        (assert (pre_op_subgoal name hello status complete))
)

;PATIENT INFORMATION
;-------------------------------------------------------------------------------------------------------------
;Some general information about the patient is needed in order to identify the specific
;case.  The following rule is used then to obtain such info and place it into a proper form
;that has been determined in Ver3.
(defrule patient_info_out
        (goal name pre_op status active)
=>
        (assert (output (type R)(format user)(variable "name,date,doctor,brainside")))
        (assert (goal interface_initiated status active))
)
```

```
(defrule patient_info_in
        (goal name pre_op status active)
        ?ans <- (answer_received (timetag ?time)
                        (text-field name ?name date ?date doctor ?doc brainside ?side)
                        (format user))
=>
        (retract ?ans)
        (assert (initial_info (name patient_name) (value ?name)))
        (assert (initial_info (name date)(value ?date)))
        (assert (initial_info (name doctor)(value ?doc)))
        (assert (initial_info (name brainside)(value ?side)))
        (assert (pre_op_subgoal name patient_info status complete))
)
```

D.7. Initialize

```
;       ---------------------------------------------------------------------------------------------------
;                       Initialize
;       Program:        Pallidotomy Version 6.0
;       Author:         Dr. N Baker and Linda Harley
;       Date:           September 2, 2003
;       File:           initial.clp
;       Updates:        Sep 2           Start reconstruction of KBS.
;                       Sep 4           Included "new_pahtplan" template to be
;                                       initialized.
;                       Oct 7           Added the calculations for depth limit wakeup.
;       ---------------------------------------------------------------------------------------------------
;PATHPLAN
;---------------------------------------------------------------------------------------------------------
;At the beginning of a track the user inputs the expected pathplan that will be found
;when going through the track.  This is merely a predicted path trajectory of the probe.
;The KBS uses the values obtained from mapping (start and end of each layer) to
;determine where we are with respect to what we were hoping to see.  Important to note
;is that the KBS will not continue unless a pathplan is provided.  Thus it is not necessary
;to request the system to send the pathplan, it will send it automatically at the begging of
;a new track.  Also remember that a different rule is fired in the interface.clp file.  We will
;always receive the OT information from the pathplan input.  It is important to note that
;this rule only refers to when the Path Plan is inputted at the beginning of a track.
;Eventually might have a rule and knowledge to update the path plan while progressing
;through the track.
```

```
(defrule request_pathplan
        (goal name initialize status active)
=>
        (assert (goal interface_initiated status active) )
        (assert (output (type R)(format model)(variable first pathplan)))
)
(defrule path_plan_in
        (goal name initialize status active)
        ?ans <- (answer_received (timetag ?time)
                        (pathplan ST ?a Le ?b GPe ?c Li ?d GPi ?e La ?f OT ?g IC ?h)
                        (format model))
=>
        (retract ?ans)
        (assert (initial_subgoal name pathplan status complete))
        (bind ?t (gensym*))
        (assert (pathplan (name ST) (start ?a) (end ?b) (track_id ?t) ) )
        (assert (pathplan (name Le) (start ?b) (end ?c) (track_id ?t) ) )
        (assert (pathplan (name GPe) (start ?c) (end ?d) (track_id ?t) ) )
        (assert (pathplan (name Li) (start ?d) (end ?e) (track_id ?t) ) )
        (assert (pathplan (name GPi) (start ?e) (end ?f) (track_id ?t) ) )
        (assert (pathplan (name La) (start ?f) (end ?g) (track_id ?t) ) )
        (assert (pathplan (name OT) (start ?g) (end ?h) (track_id ?t)))
        (assert (pathplan (name IC)(start ?h)(track_id ?t)))
        (assert (track_id ?t) )
)
```

```
;INITIALIZE TEMPLATES
;---------------------------------------------------------------------------------------------------------------
;For looping purposes it is necessary to initialize all "deftemplate" at the beginning of a
;track.  That is the purpose of this rule.
(defrule initialize_templates
        (goal name initialize status active)
        ?track <- (track_id ?t)
=>
        (retract ?track)
        (assert (control (track_id ?t)(location_id nil)(previouse_id nil)
                        (depth nil)(time 0)(layer nil)(cell nil)))
        (assert (expecting_layer (name nil)(location_id nil)(previouse_id nil)
                                 (timetag 0)))
        (assert (new_pathplan (ST 1000)(Le 1000)(GPe 1000)(Li 1000)(GPi 1000)
                              (La 1000)(OT 1000)(IC 1000)(track_id ?t)))
        (assert (wakeup_depth (ST_start nil)(ST_half nil)(ST_nend nil)(Le_half nil)
                              (GPe_start nil)(GPe_half nil)(GPe_nend nil)(Li_half nil)
                              (GPi_start nil)(GPi_half nil)(GPi_nend nil)(La_start nil)
                              (OT_start nil)(IC_start nil)(track_id ?t)(last_limit nil)
                              (status nil)))
        (assert (layer_thick (ST 1000)(GPe 1000)(GPi 1000)(track_id ?t)))
        (assert (apml_assign (ST nil)(GPe nil)(GPi nil)(OT nil)(IC nil)(status nil)
                             (track_id ?t)))
        (assert (initial_subgoal name templates status complete))
)
```

```
;CALCULATE DETH LIMITS FOR WAKEUP
;-------------------------------------------------------------------------------------------------------------------
;This following rule calculates the depths at which the KBS needs to be awakened.  The
;assignments are done in "eventwakeup" and passed on to the C++ as the depth
;increases over time.
(defrule set_depth_wakeup
        (goal name initialize status active)
        (pathplan (name ST)(start ?s1)(end ?e1)(track_id ?t1))
        (pathplan (name Le)(start ?s2)(end ?e2)(track_id ?t2))
        (pathplan (name GPe)(start ?s3)(end ?e3)(track_id ?t3))
        (pathplan (name Li)(start ?s4)(end ?e4)(track_id ?t4))
        (pathplan (name GPi)(start ?s5)(end ?e5)(track_id ?t5))
        (pathplan (name La)(start ?s6)(track_id ?t6))
        (pathplan (name OT)(start ?s7)(track_id ?t7))
        (pathplan (name IC)(start ?s8)(track_id ?t8))
        ?limit <- (wakeup_depth (ST_start nil)(ST_half nil)(ST_nend nil)(Le_half nil)
                                (GPe_start nil)(GPe_half nil)(GPe_nend nil)(Li_half nil)
                                (GPi_start nil)(GPi_half nil)(GPi_nend nil)
                                (La_start nil)(OT_start nil)(IC_start nil)
                                (track_id ?t10)(last_limit nil)(status nil))
        (test (eq ?t1 ?t2 ?t3 ?t4 ?t5 ?t6 ?t10))
=>
        (retract ?limit)
        (assert (wakeup_depth
                        (ST_start ?s1)
```

```
                (ST_half (+ ?s1 (/ (- ?e1 ?s1) 2)))
                (ST_nend (- ?e1 1))
                (Le_half (+ ?s2 (/ (- ?e2 ?s2) 2)))
                (GPe_start ?s3)
                (GPe_half (+ ?s3 (/ (- ?e3 ?s3) 2)))
                (GPe_nend (- ?e3 1))
                (Li_half (+ ?s4 (/ (- ?e4 ?s4) 2)))
                (GPi_start ?s5)
                (GPi_half (+ ?s5 (/ (- ?e5 ?s5) 2)))
                (GPi_nend (- ?e5 1))
                (La_start ?s6)
                (OT_start ?s7)
                (IC_start ?s8)
                (track_id ?t10)(last_limit nil)(status done)))
        (assert (initial_subgoal name calculate_wakeup status complete))
)
```

D.8. Track Execution

```
;       ---------------------------------------------------------------------------------------------------
;                       One Track
;       Program:        Pallidotomy Version 6.0
;       Author:         Linda Harley and Nelson Baker
;       Date:           September 1, 2003
;       File:           onetrack.clp
;       Updates:        Sep 1           Starting reconstruction of KBS.
;       ---------------------------------------------------------------------------------------------------
;FORWARD-CHAINING
;---------------------------------------------------------------------------------------------------------
;It is important to know that Onetrack is an internal loop that will continue functioning
;while the user is within a particular track.  Inputs are the pathplan, current z-depth value,
;user info and dsp.  Using these 4 inputs we infer the layer the probe is in.  The KBS one
;track is 'awaken' when a new event occurs and terminated when user selects end track.
;The "determine..." rule makes sure that all four inputs have been analyzed or looked at
;before inferring the layer.
(defrule determine_Infer_Layer
        (goal name onetrack status active)
        ?event <- (subgoal name event_wakeup status complete)
        ?depth <- (subgoal name depth status complete)
        ?dsp <- (subgoal name cell status complete)
        ?user <- (subgoal name user status complete)
=>
        (retract ?event)
```

```
        (retract ?depth)
        (retract ?dsp)
        (retract ?user)
        (assert (subgoal name infer_layer status active) )
)

;The "done_..." rules are used to set up the LHS of the rules for identifying when a
;certain phase in the subsequence has been completed, and the KBS is ready to move
;on into the next level of processing.
(defrule done_event_wakeup
        (goal name onetrack status active)
        ?event <- (subgoal name event_wakeup status active)
        ?info <- (event_subgoal name got_info status complete)
        ?wakeup <- (event_subgoal name set_wakeup status complete)
=>
        (retract ?event ?wakeup)
        (retract ?info)
        (assert (subgoal name event_wakeup status complete) )
        (assert (subgoal name depth status active) )
        (assert (subgoal name cell status active) )
        (assert (subgoal name user status active) )
)
(defrule done_depth
        (goal name onetrack status active)
        ?depth <- (subgoal name depth status active)
```

```
        ?id <- (depthsubgoal name depth_layer status complete)
        ?cell <- (depthsubgoal name cellanat status complete)
=>
        (retract ?depth)
        (retract ?id ?cell)
        (assert (subgoal name depth status complete) )
)
(defrule done_cell
        (goal name onetrack status active)
        ?dsp <- (subgoal name cell status active)
        ?done <- (cellgoal name assign status complete)
=>
        (retract ?dsp ?done)
        (assert (subgoal name cell status complete) )
)
(defrule done_user
        (goal name onetrack status active)
        ?user <- (subgoal name user status active)
        ?input <- (usergoal name user_input status complete)
=>
        (retract ?user)
        (retract ?input)
        (assert (subgoal name user status complete) )
)
(defrule done_infer_layer
```

```
        (goal name onetrack status active)
        ?layergoal <- (subgoal name infer_layer status active)
        ?infercell <- (inferlayergoal name infercell status complete)
        ?cellcleanup <- (inferlayergoal name cellcleanup status complete)
        ?celltolayer <- (inferlayergoal name celltolayer status complete)
        ?deter <- (inferlayergoal name determine_layer status complete)
        ?path <- (inferlayergoal name done_pathplan status complete)
        (control (name done)(location_id ?l1)(previouse_id ?p1)
                (track_id ?t1)(depth ?d)(time ?time2)(layer ?layer)(cell ?cell1)
                (state ?state))
=>
        (retract ?layergoal)
        (retract ?deter ?infercell ?cellcleanup ?celltolayer ?path)
        (assert (output (type D)(format data)
                        (variable Layer ?layer Cell ?cell1 State ?state)
                        (depth ?d)(timetag ?time2)))
        (assert (goal interface_initiated status active))
)
(defrule end_track
        (goal name onetrack status active)
        ?answer <- (end track)
        (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                        (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                        (GPi ?gpi)(GPistate ?gpis)(La ?la)(Lastate ?las)
                        (OT ?ot)(OTstate ?ots)(IC ?ic)(ICstate ?ics)
```

```
                        (track_id ?t1))
=>
        (retract ?answer)
        (assert (done onetrack) )
        (printout wdialog "D:KBS Pathplan*ST "?st" "?sts" Le "?le" "?les" GPe "?gpe"
"?gpes" Li "?li" "?lis" GPi "?gpi" "?gpis" La  "?la" "?las" OT "?ot" "?ots" IC "?ic"
"?ics"*!"crlf)
)
```

D.9. Event Wake Up

```
;       ---------------------------------------------------------------------------------------------------
;                       Event Wakeup
;       Program:        Pallidotomy Version 6.0
;       Author:         Linda Harley
;       Date:           September 2, 2003:
;       File:           eventwakeup.clp
;       Updates:        Sep 2, 2003:   Started reconstruction of KBS.
;                       Sep 4, 2003:   Included new defrules for end track, new track, end
;                                      op.
;                       Sep 11, 2003: Added rule depth_dsp_event
;                       Oct 7, 2003:   Added the rules dealing with depth limit wakeup.
;       ---------------------------------------------------------------------------------------------------
;DEPTH ONLY
;---------------------------------------------------------------------------------------------------
;One thing we know for certain is that the z depth value will always be available to
;reason with.  All other information might be absent but we can reason with only the
;depth value.  This first rule then places the current-depth value into the right format to
;be processed with later.
(defrule event_depth_only
        (subgoal name event_wakeup status active)
        ?answer <- (answer_received (timetag ?time1)(current-depth ?d)(format depth))
        (control (name ?name)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d2)(time ?time2)(layer ?lay1)(cell ?cell1))
        (expecting_layer (name ?name2)(location_id ?l2)(previouse_id ?p2)
```

```
                        (timetag ?time3))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)(GPe_start ?gpes)
                                (GPe_half ?gpeh)(GPe_nend ?gpee)(Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)
                                (GPi_nend ?gpie)(La_start ?las)(OT_start ?ots)
                                (IC_start ?ics)(track_id ?t3)(last_limit ?lim)(status done))
        (test (eq ?t1 ?t3))
        (test (< ?time2 ?time1))
        (test (= ?time2 ?time3))
=>
        (retract ?answer ?limit)
        (bind ?x1 (gensym*))
        (assert (control (name current)(location_id ?x1)(previouse_id ?l1)(track_id ?t1)
                        (depth ?d)(time ?time1)(layer nil)(cell nil)))
        (assert (expecting_layer (name nil)(location_id ?x1)(previouse_id ?l2)
                                (timetag ?time1)))
        (assert (deciding_layer (depth nil)(cell nil)(user nil)(decision nil)
                                (location_id ?x1)(timetag ?time1)))
        (assert (deciding_cellid_layer (anat nil)(dsp nil)(user nil)(prevhist nil)
                                        (choice nil)(override nil)(location_id ?x1)
                                        (timetag ?time1)))
        (assert (event_subgoal name got_info status complete))
        (assert (no dsp input))
        (assert (no user input))
```

```
        (assert (depth only))
        (assert (depth limit wakeup))
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                              (Le_half ?leh)(GPe_start ?gpes)
                              (GPe_half ?gpeh)(GPe_nend ?gpee)(Li_half ?lih)
                              (GPi_start ?gpis)(GPi_half ?gpih)
                              (GPi_nend ?gpie)(La_start ?las)(OT_start ?ots)
                              (IC_start ?ics)(track_id ?t3)(last_limit ?lim)(status active)))
)

;DEPTH AND DSP
;-------------------------------------------------------------------------------------------------------------
;This will only happen when KBS is awakened because the depth limit has been
;reached.
(defrule event_depth_dsp
        (subgoal name event_wakeup status active)
        ?answer <- (answer_received (timetag ?time1)(current-depth ?d)
                                    (dsp-cell sn_ratio ?sn background ?bg
                                     caudate ?ca popcorn ?pop
                                     HFD-P ?p LFD-B ?b border ?bor tonic ?ton
                                     bursting ?burst chugging ?chug pausing ?paus
                                     fiber ?fib multiple ?mul artificial ?art
                                     injury ?in tremor ?trem neg ?n)(format dsp))
        (control (name ?name)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                 (depth ?d1)(time ?time2)(layer ?lay1)(cell ?cell1))
```

```
        (expecting_layer (name ?name2)(location_id ?l2)(previouse_id ?p2)
                         (timetag ?time3))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)(GPe_start ?gpes)
                                (GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)(GPi_start ?gpis)(GPi_half ?gpih)
                                (GPi_nend ?gpie)(La_start ?las)(OT_start ?ots)
                                (IC_start ?ics)(track_id ?t3)(last_limit ?lim)(status done))
        (test (eq ?t1 ?t3))
        (test (< ?time2 ?time1))
        (test (= ?time2 ?time3))
=>
        (retract ?answer ?limit)
        (bind ?x1 (gensym*))
        (assert (control (name current)(location_id ?x1)(previouse_id ?l1)(track_id ?t1)
                         (depth ?d)(time ?time1)(layer nil)(cell nil)))
        (assert (expecting_layer (name nil)(location_id ?x1)(previouse_id ?l2)
                                 (timetag ?time1)))
        (assert (deciding_layer (depth nil)(cell nil)(user nil)(decision nil)
                                (location_id ?x1)(timetag ?time1)))
        (assert (deciding_cellid_layer (anat nil)(dsp nil)(user nil)(prevhist nil)
                                       (choice nil)(override nil)(location_id ?x1)
                                       (timetag ?time1)))
        (assert (dsp_max (max nil)(2max nil)(3max nil)(status nil)(location_id ?x1)
                         (timetag ?time1)))
```

```
        (assert (dsp_list (sn_ratio ?sn)(background ?bg)(caudate ?ca)
                          (popcorn ?pop)(HFD-P ?p)
                          (LFD-B ?b)(border ?bor)(tonic ?ton)(bursting ?burst)
                          (chugging ?chug)(pausing ?paus)(fiber ?fib)(multiple ?mul)
                          (artificial ?art)(injury ?in)(tremor ?trem)(neg ?n)
                          (location_id ?x1)(timetag ?time1)))
        (assert (event_subgoal name got_info status complete))
        (assert (no user input))
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                              (Le_half ?leh)(GPe_start ?gpes)
                              (GPe_half ?gpeh)(GPe_nend ?gpee)(Li_half ?lih)
                              (GPi_start ?gpis)(GPi_half ?gpih)
                              (GPi_nend ?gpie)(La_start ?las)(OT_start ?ots)
                              (IC_start ?ics)(track_id ?t3)(last_limit ?lim)(status active)))
)

;DEPTH, DSP AND USER
;-------------------------------------------------------------------------------------------------------------
;This rule fires whenever the user has input, dsp detected and depth available.
(defrule event_depth_dsp_user
        (subgoal name event_wakeup status active)
        ?answer <- (answer_received (timetag ?time1)(current-depth ?d)
                                    (dsp-cell sn_ratio ?sn background ?bg
                                     caudate ?ca  popcorn ?pop
                                     HFD-P ?p LFD-B ?b border ?bor tonic ?ton
```

```
                                    bursting ?burst chugging ?chug pausing ?paus
                                    fiber ?fib multiple ?mul artificial ?art
                                    injury ?in tremor ?trem neg ?n)
                                   (user layer ?lay cell ?cell)(format user))
        (control (name ?name)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                 (depth ?d1)(time ?time2)(layer ?lay1)(cell ?cell1))
        (expecting_layer (name ?name2)(location_id ?l2)(previouse_id ?p2)
                         (timetag ?time3))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                    (Le_half ?leh)(GPe_start ?gpes)
                                    (GPe_half ?gpeh)(GPe_nend ?gpee)(Li_half ?lih)
                                   (GPi_start ?gpis)(GPi_half ?gpih)
                                    (GPi_nend ?gpie)(La_start ?las)(OT_start ?ots)
                                    (IC_start ?ics)(track_id ?t3)(last_limit ?lim)(status done))
        (test (eq ?t1 ?t3))
        (test (< ?time2 ?time1))
        (test (= ?time2 ?time3))
=>
        (retract ?answer ?limit)
        (bind ?x1 (gensym*))
        (assert (control (name current)(location_id ?x1)(previouse_id ?l1)(track_id ?t1)
                         (depth ?d)(time ?time1)(layer nil)(cell nil)))
        (assert (expecting_layer (name nil)(location_id ?x1)(previouse_id ?l2)
                                    (timetag ?time1)))
        (assert (deciding_layer (depth nil)(cell nil)(user nil)(decision nil)
```

```
                    (location_id ?x1)(timetag ?time1)))
(assert (deciding_cellid_layer (anat nil)(dsp nil)(user nil)(prevhist nil)
                    (choice nil)(override nil)(location_id ?x1)
                    (timetag ?time1)))
(assert (dsp_max (max nil)(2max nil)(3max nil)(status nil)(location_id ?x1)
                    (timetag ?time1)))
(assert (dsp_list (sn_ratio ?sn)(background ?bg)(caudate ?ca)
                    (popcorn ?pop)(HFD-P ?p)
                    (LFD-B ?b)(border ?bor)(tonic ?ton)(bursting ?burst)
                    (chugging ?chug)(pausing ?paus)(fiber ?fib)(multiple ?mul)
                    (artificial ?art)(injury ?in)(tremor ?trem)(neg ?n)
                    (location_id ?x1)(timetag ?time1)))
(assert (user (layer ?lay)(cell ?cell)(location_id ?x1)(timetag ?time1)))
(assert (event_subgoal name got_info status complete))
(assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                    (Le_half ?leh)(GPe_start ?gpes)
                    (GPe_half ?gpeh)(GPe_nend ?gpee)(Li_half ?lih)
                    (GPi_start ?gpis)(GPi_half ?gpih)
                    (GPi_nend ?gpie)(La_start ?las)(OT_start ?ots)
                    (IC_start ?ics)(track_id ?t3)(last_limit ?lim)(status active)))
)
```

```
;END TRACK, NEW TRACK, AND END OP
;---------------------------------------------------------------------------------------------------------------
;Rules used to trigger the end track, new track and end op defrules.
(defrule event_end_track
        ?answer <- (answer_received (timetag ?time1)(status endtrack))
=>
        (retract ?answer)
        (assert (end track))
)
(defrule event_new_track
        ?answer <- (answer_received (timetag ?time1)(status newtrack))
=>
        (retract ?answer)
        (assert (new track))
)
(defrule event_end_op
        ?answer <- (answer_received (timetag ?time1)(status endop))
=>
        (retract ?answer)
        (assert (end operation))
)
```

```
;SET NEW DEPTH LIMIT WAKEUP
;-------------------------------------------------------------------------------------------------------------
;To ensure that the KBS gets awakened at certain points along the track for quality
;control purposes.  These following rules manage that wakeup system.  The KBS is
;woken at the beginning of a Primary Structure, halfway through a structure, and at
;1mm prior to end of Primary Structure.
(defrule set_wakeup_STstart
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (< ?d ?sts))
        (test (and (neq ?sts ?lim)
                (neq ?sth ?lim)
                (neq ?ste ?lim)
                (neq ?leh ?lim)
                (neq ?gpes ?lim)
                (neq ?gpeh ?lim)
```

```
                    (neq ?gpee ?lim)
                    (neq ?lih ?lim)
                    (neq ?gpis ?lim)
                    (neq ?gpih ?lim)
                    (neq ?gpie ?lim)
                    (neq ?las ?lim)
                    (neq ?ots ?lim)
                    (neq ?ics ?lim)))
=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?sts)(status done)))
        (assert (output (type D)(format wakeup)(variable ?sts)(text-field nil)
                        (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
(defrule set_wakeup_SThalf
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
```

```
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (and (>= ?d ?sts)
                   (< ?d ?sth)))
        (test (and (neq ?sth ?lim)
                   (neq ?ste ?lim)
                   (neq ?leh ?lim)
                   (neq ?gpes ?lim)
                   (neq ?gpeh ?lim)
                   (neq ?gpee ?lim)
                   (neq ?lih ?lim)
                   (neq ?gpis ?lim)
                   (neq ?gpih ?lim)
                   (neq ?gpie ?lim)
                   (neq ?las ?lim)
                   (neq ?ots ?lim)
                   (neq ?ics ?lim)))
=>
        (retract ?limit)
```

```
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?sth)(status done)))
        (assert (output (type D)(format wakeup)(variable ?sth)(text-field nil)
                                (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
(defrule set_wakeup_STnend
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (and (>= ?d ?sth)
                (< ?d ?ste)))
```

```
        (test (and (neq ?ste ?lim)
                   (neq ?leh ?lim)
                   (neq ?gpes ?lim)
                   (neq ?gpeh ?lim)
                   (neq ?gpee ?lim)
                   (neq ?lih ?lim)
                   (neq ?gpis ?lim)
                   (neq ?gpih ?lim)
                   (neq ?gpie ?lim)
                   (neq ?las ?lim)
                   (neq ?ots ?lim)
                   (neq ?ics ?lim)))
=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                              (Le_half ?leh)
                              (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                              (Li_half ?lih)
                              (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                              (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                              (track_id ?t2)(last_limit ?ste)(status done)))
        (assert (output (type D)(format wakeup)(variable ?ste)(text-field nil)
                              (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
```

```
(defrule set_wakeup_Lehalf
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (and (>= ?d ?ste)
                (< ?d ?leh)))
        (test (and (neq ?leh ?lim)
                (neq ?gpes ?lim)
                (neq ?gpeh ?lim)
                (neq ?gpee ?lim)
                (neq ?lih ?lim)
                (neq ?gpis ?lim)
                (neq ?gpih ?lim)
                (neq ?gpie ?lim)
                (neq ?las ?lim)
                (neq ?ots ?lim)
                (neq ?ics ?lim)))
```

```
=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?leh)(status done)))
        (assert (output (type D)(format wakeup)(variable ?leh)(text-field nil)
                                (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
(defrule set_wakeup_GPestart
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
```

```
        (test (and (>= ?d ?leh)
                   (< ?d ?gpes)))
        (test (and (neq ?gpes ?lim)
                   (neq ?gpeh ?lim)
                   (neq ?gpee ?lim)
                   (neq ?lih ?lim)
                   (neq ?gpis ?lim)
                   (neq ?gpih ?lim)
                   (neq ?gpie ?lim)
                   (neq ?las ?lim)
                   (neq ?ots ?lim)
                   (neq ?ics ?lim)))
=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                              (Le_half ?leh)
                              (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                              (Li_half ?lih)
                              (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                              (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                              (track_id ?t2)(last_limit ?gpes)(status done)))
        (assert (output (type D)(format wakeup)(variable ?gpes)(text-field nil)
                              (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
```

```
(defrule set_wakeup_GPehalf
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (and (>= ?d ?gpes)
                (< ?d ?gpeh)))
        (test (and (neq ?gpeh ?lim)
                (neq ?gpee ?lim)
                (neq ?lih ?lim)
                (neq ?gpis ?lim)
                (neq ?gpih ?lim)
                (neq ?gpie ?lim)
                (neq ?las ?lim)
                (neq ?ots ?lim)
                (neq ?ics ?lim)))
=>
        (retract ?limit)
```

```
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?gpeh)(status done)))
        (assert (output (type D)(format wakeup)(variable ?gpeh)(text-field nil)
                                (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
(defrule set_wakeup_GPenend
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (and (>= ?d ?gpeh)
                (< ?d ?gpee)))
```

```
        (test (and (neq ?gpee ?lim)
                   (neq ?lih ?lim)
                   (neq ?gpis ?lim)
                   (neq ?gpih ?lim)
                   (neq ?gpie ?lim)
                   (neq ?las ?lim)
                   (neq ?ots ?lim)
                   (neq ?ics ?lim)))
=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                              (Le_half ?leh)
                              (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                              (Li_half ?lih)
                              (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                              (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                              (track_id ?t2)(last_limit ?gpee)(status done)))
        (assert (output (type D)(format wakeup)(variable ?gpee)(text-field nil)
                              (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
```

```
(defrule set_wakeup_Lihalf
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (and (>= ?d ?gpee)
                (< ?d ?lih)))
        (test (and (neq ?lih ?lim)
                (neq ?gpis ?lim)
                (neq ?gpih ?lim)
                (neq ?gpie ?lim)
                (neq ?las ?lim)
                (neq ?ots ?lim)
                (neq ?ics ?lim)))
=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
```

```
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lih)(status done)))
        (assert (output (type D)(format wakeup)(variable ?lih)(text-field nil)
                                (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
(defrule set_wakeup_GPistart
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (and (>= ?d ?lih)
                (< ?d ?gpis)))
        (test (and (neq ?gpis ?lim)
                (neq ?gpih ?lim)
```

```
                    (neq ?gpie ?lim)
                    (neq ?las ?lim)
                    (neq ?ots ?lim)
                    (neq ?ics ?lim)))
=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?gpis)(status done)))
        (assert (output (type D)(format wakeup)(variable ?gpis)(text-field nil)
                                (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
(defrule set_wakeup_GPihalf
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
```

```
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (and (>= ?d ?gpis)
                   (< ?d ?gpih)))
        (test (and (neq ?gpih ?lim)
                   (neq ?gpie ?lim)
                   (neq ?las ?lim)
                   (neq ?ots ?lim)
                   (neq ?ics ?lim)))
=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?gpih)(status done)))
        (assert (output (type D)(format wakeup)(variable ?gpih)(text-field nil)
                                (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
```

```
(defrule set_wakeup_GPinend
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (and (>= ?d ?gpih)
                (< ?d ?gpie)))
        (test (and (neq ?gpie ?lim)
                (neq ?las ?lim)
                (neq ?ots ?lim)
                (neq ?ics ?lim)))
=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
```

```
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?gpie)(status done)))
        (assert (output (type D)(format wakeup)(variable ?gpie)(text-field nil)
                                (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
(defrule set_wakeup_Lastart
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (and (>= ?d ?gpie)
                (< ?d ?las)))
        (test (and (neq ?las ?lim)
                (neq ?ots ?lim)
                (neq ?ics ?lim)))
=>
        (retract ?limit)
```

```
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?gpie)(status done)))
        (assert (output (type D)(format wakeup)(variable ?las)(text-field nil)
                                (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
(defrule set_wakeup_La_OT_warning
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (>= ?d ?las))
        (test (< ?d ?ots))
```

```
        (test (neq ?ots 1000))
        (test (and (neq ?las ?lim)
                (neq ?ics ?lim)))
=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?las)(status done)))
        (printout wdialog "M:message*nil*Warning: Probe close to the Optic Track and
Intenral Capsule.  Proceed with caution.!" crlf)
        (assert (output (type D)(format wakeup)(variable ?las)(text-field nil)
                        (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
(defrule set_wakeup_La_IC_special
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
```

```
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (>= ?d ?las))
        (test (< ?d ?ics))
        (test (eq ?ots 1000))
        (test (neq ?las ?lim))

=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?las)(status done)))
        (printout wdialog "M:message*nil*Warning: Probe close to Optic Track and
Internal Capsule.  Proceed with caution.!" crlf)
        (assert (output (type D)(format wakeup)(variable ?las)(text-field nil)
                        (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
```

```
(defrule set_wakeup_OT_warning
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (>= ?d ?ots))
        (test (< ?d ?ics))
        (test (and (neq ?ots ?lim)
                (neq ?ics ?lim)))
=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?ots)(status done)))
```

```
        (printout wdialog "M:message*nil*Warning:  Probe entering Optic Track.  Proceed
with utmost caution.!" crlf)
        (assert (output (type D)(format wakeup)(variable ?las)(text-field nil)
                        (depth ?d)(timetag ?time1)))
        (assert (goal set_wakeup status complete))
)
(defrule set_wakeup_IC_warning
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                (Li_half ?lih)
                                (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                (track_id ?t2)(last_limit ?lim)(status active))
        (test (eq ?t1 ?t2))
        (test (>= ?d ?ics))
        (test (neq ?ics ?lim))
=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                (Le_half ?leh)
                                (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
```

```
                                    (Li_half ?lih)
                                    (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                    (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                    (track_id ?t2)(last_limit ?ics)(status done)))
        (assert (output (type D)(format wakeup)(variable ?las)(text-field nil)
                        (depth ?d)(timetag ?time1)))
        (printout wdialog "M:message*nil*Warning: Probe enetering Internal Capsule.
Proceed with caution.!")
        (assert (goal set_wakeup status complete))
)
(defrule set_wakeup_nothing
        (subgoal name event_wakeup status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d)(time ?time1)(layer nil)(cell nil))
        ?limit <- (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                                    (Le_half ?leh)
                                    (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                                    (Li_half ?lih)
                                    (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                                    (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                                    (track_id ?t2)(last_limit ?lim)(status active))
        (test(eq ?t1 ?t2))
        (test (or       (eq ?lim ?sts)
                        (eq ?lim ?sth)
                        (eq ?lim ?ste)
```

```
                    (eq ?lim ?leh)
                    (eq ?lim ?gpes)
                    (eq ?lim ?gpeh)
                    (eq ?lim ?gpee)
                    (eq ?lim ?lih)
                    (eq ?lim ?gpis)
                    (eq ?lim ?gpih)
                    (eq ?lim ?gpie)
                    (eq ?lim ?las)
                    (eq ?lim ?ots)
                    (eq ?lim ?ics)))
=>
        (retract ?limit)
        (assert (wakeup_depth (ST_start ?sts)(ST_half ?sth)(ST_nend ?ste)
                              (Le_half ?leh)
                              (GPe_start ?gpes)(GPe_half ?gpeh)(GPe_nend ?gpee)
                              (Li_half ?lih)
                              (GPi_start ?gpis)(GPi_half ?gpih)(GPi_nend ?gpie)
                              (La_start ?las)(OT_start ?ots)(IC_start ?ics)
                              (track_id ?t2)(last_limit ?lim)(status done)))
        (assert (event_subgoal name set_wakeup status complete))
)
```

D.10. Depth

```
;       ---------------------------------------------------------------------------------------------------
;                       Depth
;       Program:        Pallidotomy Version 6.0
;       Author:         Linda Harley and Nelson Baker
;       Date:           September 2,2003
;       File:           depth.clp
;       Updates:        Sept 2, 2003    Starting reconstruction of KBS.
;                       Sept 4, 2003    Removed retract of "first time"
;                       Oct 8, 2003     Added "determine_before_Striatum" and
;                                       "Striatum_before" rules.
;       ---------------------------------------------------------------------------------------------------
;STRIATUM (ST)
;---------------------------------------------------------------------------------------------------------
;Using the pathplan provided we now compare the current z depth against this
;pathplan to determine in what predicted layer the probe is suppose to be.
(defrule determine_before_Striatum
        (subgoal name depth status active)
        (pathplan (name ST)(start ?s)(end ?e)(track_id ?t1))
        (control (name current)(location_id ?l2)(previouse_id ?p2)
                (depth ?d)(track_id ?t2)(time ?time2))
        ?expect <- (expecting_layer (name nil)(location_id ?l3)(timetag ?time3))
        (test (and (eq ?l3 ?l2)
                  (eq ?t1 ?t2)))
        (test (< ?d ?s))
```

```
=>
        (retract ?expect)
        (assert (expecting_layer (name none)(location_id ?l3)(timetag ?time3)))
)
(defrule Striatum_before
        (expecting_layer (name none)(location_id ?l1)(timetag ?time1))
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)
                                   (decision ?dec)(location_id ?l2)
                                   (timetag ?time2))
        (test (eq ?time1 ?time2))
=>
        (retract ?decide)
        (assert (deciding_layer (depth none)(cell ?dsp)(user ?user)
                                  (decision ?dec)(location_id ?l2)(timetag ?time2)))
        (assert (depthsubgoal name depth_layer status complete))
)
(defrule determine_Striatum
        (subgoal name depth status active)
        (pathplan (name ST) (start ?s) (end ?e) (track_id ?t1) )
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (track_id ?t2) (time ?time1))
        ?expect <- (expecting_layer (name nil) (location_id ?l4) (timetag ?time2) )
        (test (eq ?l4 ?l1) )
        (test (eq ?t1 ?t2) )
        (test (and (>= ?c ?s) (< ?c ?e) ) )
```

```
=>
        (retract ?expect)
        (assert (expecting_layer (name ST) (location_id ?l4) (timetag ?time2)) )
)
(defrule Striatum_downward
        (control (name current)(location_id ?l1)(previouse_id ?l2)(depth ?c)(time ?time1))
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2))
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
        (test (eq ?l2 ?l5) )
        (test (>= ?c ?d) )
        (anatomy_layer_down (previouse ST) (down ?own) )
        (test (eq ?n ?own) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
=>
        (retract ?decide)
        (assert (depthsubgoal name depth_layer status complete) )
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) ) )
)
(defrule Striatum_upward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (time ?time1))
```

```
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2))
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
        (test (eq ?l2 ?l5) )
        (test (< ?c ?d) )
        (anatomy_layer_up (previouse ST) (up ?up) )
        (test (eq ?n ?up) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                   (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
=>
        (retract ?decide)
        (assert (depthsubgoal name depth_layer status complete) )
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                   (location_id ?l9) (timetag ?time4) ) )
)

;FIRST TIME
;-------------------------------------------------------------------------------------------------------------
;The first case within a new track is always unique because you do not have something
;to compare to.  Thus the reason for this rule to deal with such a case.  This rule
;will only fire once at the beginning of a new track. NOTE: removed the retract
;of first time.  this is now done in "inferlayer.clp"
(defrule first_time
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
```

```
                (time ?time1))
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2))
        (test (eq ?l1 ?l3) )
        (anatomy_layer_down (previouse none) (down ?own) )
        (test (eq ?n ?own) )
        (first time)
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
=>
        (retract ?decide)
        (assert (depthsubgoal name depth_layer status complete) )
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) ) )
)

;LAMINA EXTERIOR (Le)
;-------------------------------------------------------------------------------------------------------------
;Using pathplan and the current z depth able to determine whether the probe is in
;the Lamina Exterior.
(defrule determine_Lamina_exterior
        (subgoal name depth status active)
        (pathplan (name Le) (start ?s) (end ?e) (track_id ?t1) )
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (track_id ?t2) (time ?time1) )
```

```
        ?expect <- (expecting_layer (name nil) (location_id ?l4) (timetag ?time2) )
        (test (eq ?l4 ?l1) )
        (test (eq ?t1 ?t2) )
        (test (and (>= ?c ?s) (< ?c ?e) ) )
=>
        (retract ?expect)
        (assert (expecting_layer (name Le) (location_id ?l4) (timetag ?time2) ) )
)
(defrule Le_downward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)(time
?time1))
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2))
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
        (test (eq ?l2 ?l5) )
        (test (>= ?c ?d) )
        (anatomy_layer_down (previouse Le) (down ?own) )
        (test (eq ?n ?own) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
=>
        (retract ?decide)
        (assert (depthsubgoal name depth_layer status complete) )
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
```

```
                                (location_id ?l9) (timetag ?time4) ) )
)
(defrule Le_upward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (time ?time1))
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2))
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)
                (time ?time3))
        (test (eq ?l2 ?l5) )
        (test (< ?c ?d) )
        (anatomy_layer_up (previouse Le) (up ?up) )
        (test (eq ?n ?up) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
=>
        (retract ?decide)
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) ) )
        (assert (depthsubgoal name depth_layer status complete) )
)
```

```
;GLOBUS PALLIDUS EXTERIOR (GPe)
;-------------------------------------------------------------------------------------------------------------
;Using the pathplan and the current z depth able to determine whether the probe
;is in the GPe.
(defrule determine_GPe
        (subgoal name depth status active)
        (pathplan (name GPe) (start ?s) (end ?e) (track_id ?t1) )
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (track_id ?t2) (time ?time1))
        ?expect <- (expecting_layer (name nil) (location_id ?l4)(timetag ?time2) )
        (test (eq ?l4 ?l1) )
        (test (eq ?t1 ?t2) )
        (test (and (>= ?c ?s) (< ?c ?e) ) )
=>
        (retract ?expect)
        (assert (expecting_layer (name GPe) (location_id ?l4) (timetag ?time2) ) )
)
(defrule GPe_downward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (time ?time1))
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2) )
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
        (test (eq ?l2 ?l5) )
        (test (>= ?c ?d) )
```

```
        (anatomy_layer_down (previouse GPe) (down ?own) )
        (test (eq ?n ?own) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                   (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
=>
        (retract ?decide)
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                  (location_id ?l9) (timetag ?time4) ) )
        (assert (depthsubgoal name depth_layer status complete) )
)
(defrule GPe_upward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                 (time ?time1))
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2))
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
        (test (eq ?l2 ?l5) )
        (test (< ?c ?d) )
        (anatomy_layer_up (previouse GPe) (up ?up) )
        (test (eq ?n ?up) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                   (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
=>
```

```
        (retract ?decide)
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) ) )
        (assert (depthsubgoal name depth_layer status complete) )
)

;LAMINA INTERIOR (Li)
;-------------------------------------------------------------------------------------------------------------
;Using the pathplan and the current z depth able to determine whether the probe
;is in the Li.
(defrule determine_Li
        (subgoal name depth status active)
        (pathplan (name Li) (start ?s) (end ?e) (track_id ?t1) )
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (track_id ?t2) (time ?time1))
        ?expect <- (expecting_layer (name nil) (location_id ?l4) (timetag ?time2))
        (test (eq ?l4 ?l1) )
        (test (eq ?t1 ?t2) )
        (test (and (>= ?c ?s) (< ?c ?e) ) )
=>
        (retract ?expect)
        (assert (expecting_layer (name Li) (location_id ?l4) (timetag ?time2)) )
)
```

```
(defrule Li_downward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (time ?time1))
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2))
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
        (test (eq ?l2 ?l5) )
        (test (>= ?c ?d) )
        (anatomy_layer_down (previouse Li) (down ?own) )
        (test (eq ?n ?own) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
=>
        (retract ?decide)
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) ) )
        (assert (depthsubgoal name depth_layer status complete) )
)
(defrule Li_upward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (time ?time1))
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2))
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
```

```
        (test (eq ?l2 ?l5) )
        (test (< ?c ?d) )
        (anatomy_layer_up (previouse Li) (up ?up) )
        (test (eq ?n ?up) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                   (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
=>
        (retract ?decide)
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                  (location_id ?l9) (timetag ?time4) ) )
        (assert (depthsubgoal name depth_layer status complete) )
)

;GLOBUS PALLIDUS INTERIOR (GPi)
;-------------------------------------------------------------------------------------------------------------
;Using pathplan and current z depth able to determine whether the probe is in the
;GPi.
(defrule determine_GPi
        (subgoal name depth status active)
        (pathplan (name GPi) (start ?s) (end ?e) (track_id ?t1) )
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (track_id ?t2) (time ?time1))
        ?expect <- (expecting_layer (name nil) (location_id ?l4) (timetag ?time2))
        (test (eq ?l1 ?l4) )
```

```
        (test (eq ?t1 ?t2) )
        (test (and (>= ?c ?s) (< ?c ?e) ) )
=>
        (retract ?expect)
        (assert (expecting_layer (name GPi) (location_id ?l4) (timetag ?time2)) )
)
(defrule GPi_downward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (time ?time1))
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2))
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
        (test (eq ?l2 ?l5) )
        (test (>= ?c ?d) )
        (anatomy_layer_down (previouse GPi) (down ?own) )
        (test (eq ?n ?own) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
        (not (do OT))
        (not (do IC))
=>
        (retract ?decide)
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) ) )
```

```
        (assert (depthsubgoal name depth_layer status complete) )
)
(defrule GPi_upward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (time ?time1))
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2))
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
        (test (eq ?l2 ?l5) )
        (test (< ?c ?d) )
        (anatomy_layer_up (previouse GPi) (up ?up) )
        (test (eq ?n ?up) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
        (not (do OT))
        (not (do IC))
=>
        (retract ?decide)
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) ) )
        (assert (depthsubgoal name depth_layer status complete) )
)
```

```
;LAMINA ANTERIOR (La)
;-----------------------------------------------------------------------------------------------------------
;Using the pathplan and the current z depth able to determine whether the probe
;is in the La.
(defrule determine_La_OT
        (subgoal name depth status active)
        (pathplan (name La) (start ?s) (end ?e) (track_id ?t1) )
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (track_id ?t2) (time ?time1))
        ?expect <- (expecting_layer (name nil) (location_id ?l4) (timetag ?time2))
        (test (eq ?l1 ?l4) )
        (test (eq ?t1 ?t2) )
        (test (>= ?c ?s)  )
        (test (< ?c ?e) )
        (test (neq ?e 1000))
=>
        (retract ?expect)
        (assert (expecting_layer (name La) (location_id ?l4) (timetag ?time2)) ) )
(defrule determine_La_IC
        (subgoal name depth status active)
        (pathplan (name La)(start ?s1)(end ?e1)(track_id ?t1))
        (pathplan (name IC)(start ?s2)(end ?e2)(track_id ?t2))
        (control (name current)(location_id ?l3)(previouse_id ?p3)(depth ?c)
                (track_id ?t3)(time ?time3))
        ?expect <- (expecting_layer (name nil)(location_id ?l4)(timetag ?time4))
```

```
        (test (eq ?l3 ?l4))
        (test (eq ?t1 ?t2 ?t3))
        (test (>= ?c ?s1))
        (test (< ?c ?s2))
        (test (eq ?e1 1000))
=>
        (retract ?expect)
        (assert (expecting_layer (name La)(location_id ?l4)(timetag ?time4)))
)
(defrule La_downward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (time ?time1))
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2))
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
        (test (eq ?l2 ?l5) )
        (test (>= ?c ?d) )
        (anatomy_layer_down (previouse La) (down ?own) )
        (test (eq ?n ?own) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
        (not (do OT))
        (not (do IC))
=>
```

```
        (retract ?decide)
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) ) )
        (assert (depthsubgoal name depth_layer status complete) )
)
(defrule La_upward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (time ?time1))
        (expecting_layer (name ?n) (location_id ?l3)(timetag ?time2) )
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
        (test (eq ?l2 ?l5) )
        (test (< ?c ?d) )
        (anatomy_layer_up (previouse La) (up ?up) )
        (test (eq ?n ?up) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
        (not (do OT))
        (not (do IC))
=>
        (retract ?decide)
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) ) )
        (assert (depthsubgoal name depth_layer status complete) ))
```

```
;OPTIC TRACK (OT)
;-------------------------------------------------------------------------------------------------------------
;Using the pathplan and the current z depth able to determine whether the probe
;is in the OT.
(defrule determine_OT
        (subgoal name depth status active)
        (pathplan (name OT) (start ?s) (end ?e) (track_id ?t1) )
        (pathplan (name IC) (start ?s2)(end ?e2)(track_id ?t3))
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (track_id ?t2) (time ?time1))
        ?expect <- (expecting_layer (name nil) (location_id ?l4) (timetag ?time2))
        (test (eq ?l1 ?l4) )
        (test (eq ?t1 ?t2 ?t3) )
        (test (>= ?c ?s)  )
        (test (< ?c ?s2))
=>
        (retract ?expect)
        (assert (do OT))  ;handler
        (assert (expecting_layer (name OT) (location_id ?l4) (timetag ?time2)) ) )
(defrule OT_downward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (time ?time1))
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2))
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
```

```
        (test (eq ?l2 ?l5) )
        (test (>= ?c ?d) )
        (anatomy_layer_down (previouse OT) (down ?own) )
        (test (eq ?n ?own) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                   (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
        ?handler <- (do OT)
=>
        (retract ?decide ?handler)
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                   (location_id ?l9) (timetag ?time4) ) )
        (assert (depthsubgoal name depth_layer status complete) )
)
(defrule OT_upward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                 (time ?time1))
        (expecting_layer (name ?n) (location_id ?l3)(timetag ?time2) )
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
        (test (eq ?l2 ?l5) )
        (test (< ?c ?d) )
        (anatomy_layer_up (previouse OT) (up ?up) )
        (test (eq ?n ?up) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
```

```
                                        (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
        ?handler <- (do OT)
=>
        (retract ?decide ?handler)
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                        (location_id ?l9) (timetag ?time4) ) )
        (assert (depthsubgoal name depth_layer status complete) )
)

;INTERNAL CAPSULE (IC)
;-------------------------------------------------------------------------------------------------------------
;Using the pathplan and the current z depth able to determine whether the probe
;is in the La.
(defrule determine_IC
        (subgoal name depth status active)
        (pathplan (name IC) (start ?s) (end ?e) (track_id ?t1) )
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (track_id ?t2) (time ?time1))
        ?expect <- (expecting_layer (name nil) (location_id ?l4) (timetag ?time2))
        (test (eq ?l1 ?l4) )
        (test (eq ?t1 ?t2) )
        (test (>= ?c ?s)  )
=>
        (retract ?expect)
```

```
        (assert (do IC))
        (assert (expecting_layer (name IC) (location_id ?l4) (timetag ?time2)) )
)
(defrule IC_downward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c
                (time ?time1))
        (expecting_layer (name ?n) (location_id ?l3) (timetag ?time2))
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
        (test (eq ?l2 ?l5) )
        (test (>= ?c ?d) )
        (anatomy_layer_down (previouse IC) (down ?own) )
        (test (eq ?n ?own) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
        ?handler <- (do IC)
=>
        (retract ?decide ?handler)
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                (location_id ?l9) (timetag ?time4) ) )
        (assert (depthsubgoal name depth_layer status complete) ) )
(defrule IC_upward
        (control (name current) (location_id ?l1) (previouse_id ?l2) (depth ?c)
                (time ?time1))
```

```
        (expecting_layer (name ?n) (location_id ?l3)(timetag ?time2) )
        (test (eq ?l1 ?l3) )
        (control (name done) (location_id ?l5) (previouse_id ?l6) (depth ?d)(time ?time3))
        (test (eq ?l2 ?l5) )
        (test (< ?c ?d) )
        (anatomy_layer_up (previouse IC) (up ?up) )
        (test (eq ?n ?up) )
        ?decide <- (deciding_layer (depth nil)(cell ?dsp)(user ?user)(decision ?dec)
                                   (location_id ?l9) (timetag ?time4) )
        (test (eq ?l1 ?l9) )
        ?handler <- (do IC)
=>
        (retract ?decide ?handler)
        (assert (deciding_layer (depth ?n)(cell ?dsp)(user ?user)(decision ?dec)
                                   (location_id ?l9) (timetag ?time4) ) )
        (assert (depthsubgoal name depth_layer status complete) )
)
```

D.11. DSP

```
;       ---------------------------------------------------------------------------------------------------
;                       DSP
;       Program:        Pallidotomy Version 6.0
;       Author:         Linda Harley
;       Date:           September 11, 2003
;       File:           dsp.clp
;       Updates:        Sep 14, 2003: Included resoning for primary/secondary.
;       ---------------------------------------------------------------------------------------------------
;SIGNAL TO NOISE RATIO
;---------------------------------------------------------------------------------------------------------
;If signal/noise is high it means probe is in primary structure (ST, GPe, GPi)
;and if signal/noise is low it means probe is in secondary structure (Lamina)
;This distinction is thus encoded in the next 2 rules.
(defrule sn_primary_structure
        (subgoal name cell status active)
        (dsp_list (sn_ratio ?sn)(background ?bg)(popcorn ?pop)(HFD-P ?p)
                (LFD-B ?b)(border ?bor)(tonic ?ton)(bursting ?burst)
                (chugging ?chug)(pausing ?paus)(fiber ?fib)(multiple ?mul)
                (artificial ?art)(injury ?in)(tremor ?trem)
                (location_id ?l1)(timetag ?time1))
        (control (name current)(location_id ?l2)(previouse_id ?p2)
                (track_id ?t2)(depth ?d)(time ?time2)(layer nil)(cell nil))
        (test (eq ?l2 ?l1 ))
        (test (> ?sn 15))       ;note 15 is experimental value
```

```
=>
        (assert (cellgoal name primary status active)) )
(defrule sn_secondary_structure
        (subgoal name cell status active)
        (dsp_list (sn_ratio ?sn)(background ?bg)(popcorn ?pop)(HFD-P ?p)
                (LFD-B ?b)(border ?bor)(tonic ?ton)(bursting ?burst)
                (chugging ?chug)(pausing ?paus)(fiber ?fib)(multiple ?mul)
                (artificial ?art)(injury ?in)(tremor ?trem)
                (location_id ?l1)(timetag ?time1))
        (control (name current)(location_id ?l2)(previouse_id ?p2)
                (track_id ?t2)(depth ?d)(time ?time2)(layer nil)(cell nil))
        (test (eq ?l2 ?l1))
        (test (<= ?sn 15))       ;note 15 is experimental value
=>
        (assert (cellgoal name secondary status active))
)

;FUNCTIONS
;-----------------------------------------------------------------------------------------------------------------
;The following are the functions use to do the math/reasoning behind selecting
;the maximum 3 dsp cell types.
;FIND MAX finds the maximum value in the list of cell probs sent to it.

(deffunction find_max (?pop ?p ?b ?bor ?ton ?burst ?chug ?paus ?fib ?mul ?art ?in
?trem)
```

```
        (max ?pop ?p ?b ?bor ?ton ?burst ?chug ?paus ?fib ?mul ?art ?in ?trem)
)

;REPLACE.  This rule is responsible for normalizing the previous max. type
;to zero and setting it up so that the maxvalue can be found. It serves as a
;replacement type of function
(deffunction replace (?maxvalue ?sn ?bg ?pop ?p ?b ?bor ?ton ?burst ?chug ?paus ?fib
                      ?mul ?art ?in ?trem ?l1 ?time1)
  (if (= ?maxvalue ?pop) then (assert (dsp_list (sn_ratio ?sn)(background ?bg)
                                                (popcorn 0)(HFD-P ?p)(LFD-B ?b)
                                                (border ?bor)(tonic ?ton)
                                                (bursting ?burst)(chugging ?chug)
                                                (pausing ?paus)(fiber ?fib)
                                                (multiple ?mul)(artificial ?art)
                                                (injury ?in)(tremor ?trem)
                                                (location_id ?l1)(timetag ?time1)))
else
  (if (= ?maxvalue ?p) then (assert (dsp_list (sn_ratio ?sn)(background ?bg)
                                                (popcorn ?pop)(HFD-P 0)(LFD-B ?b)
                                                (border ?bor)(tonic ?ton)
                                                (bursting ?burst)(chugging ?chug)
                                                (pausing ?paus)(fiber ?fib)
                                                (multiple ?mul)(artificial ?art)
                                                (injury ?in)(tremor ?trem)
                                                (location_id ?l1)(timetag ?time1)))
```

```
else
 (if (= ?maxvalue ?b) then (assert (dsp_list (sn_ratio ?sn)(background ?bg)
                                             (popcorn ?pop)(HFD-P ?p)(LFD-B 0)
                                             (border ?bor)(tonic ?ton)
                                             (bursting ?burst)(chugging ?chug)
                                             (pausing ?paus)(fiber ?fib)
                                             (multiple ?mul)(artificial ?art)
                                             (injury ?in)(tremor ?trem)
                                             (location_id ?l1)(timetag ?time1)))
else
 (if (= ?maxvalue ?bor) then (assert (dsp_list (sn_ratio ?sn)(background ?bg)
                                             (popcorn ?pop)(HFD-P ?p)(LFD-B ?b)
                                             (border 0)(tonic ?ton)
                                             (bursting ?burst)(chugging ?chug)
                                             (pausing ?paus)(fiber ?fib)
                                             (multiple ?mul)(artificial ?art)
                                             (injury ?in)(tremor ?trem)
                                             (location_id ?l1)(timetag ?time1)))
else
 (if (= ?maxvalue ?ton) then (assert (dsp_list (sn_ratio ?sn)(background ?bg)
                                             (popcorn ?pop)(HFD-P ?p)(LFD-B ?b)
                                             (border ?bor)(tonic 0)
                                             (bursting ?burst)(chugging ?chug)
                                             (pausing ?paus)(fiber ?fib)
                                             (multiple ?mul)(artificial ?art)
```

```
                                        (injury ?in)(tremor ?trem)
                                        (location_id ?l1)(timetag ?time1)))
else
 (if (= ?maxvalue ?burst) then (assert (dsp_list (sn_ratio ?sn)(background ?bg)
                                        (popcorn ?pop)(HFD-P ?p)(LFD-B ?b)
                                        (border ?bor)(tonic ?ton)
                                        (bursting 0)(chugging ?chug)
                                        (pausing ?paus)(fiber ?fib)
                                        (multiple ?mul)(artificial ?art)
                                        (injury ?in)(tremor ?trem)
                                        (location_id ?l1)(timetag ?time1)))
else
 (if (= ?maxvalue ?chug) then (assert (dsp_list (sn_ratio ?sn)(background ?bg)
                                        (popcorn ?pop)(HFD-P ?p)(LFD-B ?b)
                                        (border ?bor)(tonic ?ton)
                                        (bursting ?burst)(chugging 0)
                                        (pausing ?paus)(fiber ?fib)
                                        (multiple ?mul)(artificial ?art)
                                        (injury ?in)(tremor ?trem)
                                        (location_id ?l1)(timetag ?time1)))
else
 (if (= ?maxvalue ?paus) then (assert (dsp_list (sn_ratio ?sn)(background ?bg)
                                        (popcorn ?pop)(HFD-P ?p)(LFD-B ?b)
                                        (border ?bor)(tonic ?ton)
                                        (bursting ?burst)(chugging ?chug)
```

```
                                        (pausing 0)(fiber ?fib)
                                        (multiple ?mul)(artificial ?art)
                                        (injury ?in)(tremor ?trem)
                                        (location_id ?l1)(timetag ?time1)))
else
 (if (= ?maxvalue ?fib) then (assert (dsp_list (sn_ratio ?sn)(background ?bg)
                                        (popcorn ?pop)(HFD-P ?p)(LFD-B ?b)
                                        (border ?bor)(tonic ?ton)
                                        (bursting ?burst)(chugging ?chug)
                                        (pausing ?paus)(fiber 0)
                                        (multiple ?mul)(artificial ?art)
                                        (injury ?in)(tremor ?trem)
                                        (location_id ?l1)(timetag ?time1)))
else
 (if (= ?maxvalue ?mul) then (assert (dsp_list (sn_ratio ?sn)(background ?bg)
                                        (popcorn ?pop)(HFD-P ?p)(LFD-B ?b)
                                        (border ?bor)(tonic ?ton)
                                        (bursting ?burst)(chugging ?chug)
                                        (pausing ?paus)(fiber ?fib)
                                        (multiple 0)(artificial ?art)
                                        (injury ?in)(tremor ?trem)
                                        (location_id ?l1)(timetag ?time1)))
else
 (if (= ?maxvalue ?art) then (assert (dsp_list (sn_ratio ?sn)(background ?bg)
                                        (popcorn ?pop)(HFD-P ?p)(LFD-B ?b)
```

```
                                        (border ?bor)(tonic ?ton)
                                        (bursting ?burst)(chugging ?chug)
                                        (pausing ?paus)(fiber ?fib)
                                        (multiple ?mul)(artificial 0)
                                        (injury ?in)(tremor ?trem)
                                        (location_id ?l1)(timetag ?time1)))
else
 (if (= ?maxvalue ?in) then (assert (dsp_list (sn_ratio ?sn)(background ?bg)
                                        (popcorn ?pop)(HFD-P ?p)(LFD-B ?b)
                                        (border ?bor)(tonic ?ton)
                                        (bursting ?burst)(chugging ?chug)
                                        (pausing ?paus)(fiber ?fib)
                                        (multiple ?mul)(artificial ?art)
                                        (injury 0)(tremor ?trem)
                                        (location_id ?l1)(timetag ?time1)))
else
 (if (= ?maxvalue ?trem) then (assert (dsp_list (sn_ratio ?sn)(background ?bg)
                                        (popcorn ?pop)(HFD-P ?p)(LFD-B ?b)
                                        (border ?bor)(tonic ?ton)
                                        (bursting ?burst)(chugging ?chug)
                                        (pausing ?paus)(fiber ?fib)
                                        (multiple ?mul)(artificial ?art)
                                        (injury ?in)(tremor 0)
                                        (location_id ?l1)(timetag ?time1)))
))))))))))))))
```

```
;CELL ORDER MAX VAL.  This function is responsible for finding the maximum
;probability and associating the correct cell type with it.
(deffunction cellorder_maxval (?maxvalue ?pop ?p ?b ?bor ?ton ?burst ?chug ?paus ?fib
                               ?mul ?art ?in ?trem ?l2 ?time2)
  (if (= ?maxvalue ?pop) then (assert (dsp_max (max popcorn)(2max nil)(3max nil)
                                       (location_id ?l2)(timetag ?time2)))
else
  (if (= ?maxvalue ?p) then (assert (dsp_max (max HFD-P)(2max nil)(3max nil)
                                       (location_id ?l2)(timetag ?time2)))
else
  (if (= ?maxvalue ?b) then (assert (dsp_max (max LFD-B)(2max nil)(3max nil)
                                       (location_id ?l2)(timetag ?time2)))
else
  (if (= ?maxvalue ?bor) then (assert (dsp_max (max border)(2max nil)(3max nil)
                                       (location_id ?l2)(timetag ?time2)))
else
  (if (= ?maxvalue ?ton) then (assert (dsp_max (max tonic)(2max nil)(3max nil)
                                       (location_id ?l2)(timetag ?time2)))
else
  (if (= ?maxvalue ?burst) then (assert (dsp_max (max bursting)(2max nil)(3max nil)
                                       (location_id ?l2)(timetag ?time2)))
else
  (if (= ?maxvalue ?chug) then (assert (dsp_max (max chugging)(2max nil)(3max nil)
                                       (location_id ?l2)(timetag ?time2)))
else
```

```
 (if (= ?maxvalue ?paus) then (assert (dsp_max (max pausing)(2max nil)(3max nil)
                                        (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?fib) then (assert (dsp_max (max fiber)(2max nil)(3max nil)
                                        (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?mul) then (assert (dsp_max (max multiple)(2max nil)(3max nil)
                                        (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?art) then (assert (dsp_max (max artificial)(2max nil)(3max nil)
                                        (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?in) then (assert (dsp_max (max injury)(2max nil)(3max nil)
                                        (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?trem) then (assert (dsp_max (max tremor)(2max nil)(3max nil)
                                        (location_id ?l2)(timetag ?time2)))
))))))))))))))

;CELL ORDER 2 MAX VAL. This rule is responsible for ifnding the 2nd maximum
;probability and associating the correct cell type with it.
(deffunction cellorder_2maxval (?maxvalue ?pop ?p ?b ?bor ?ton ?burst ?chug ?paus
                                ?fib ?mul ?art ?in ?trem ?maxi ?l2 ?time2)
 (if (= ?maxvalue ?pop) then (assert (dsp_max (max ?maxi)(2max popcorn)(3max nil)
                                        (location_id ?l2)(timetag ?time2)))
```

```
else
 (if (= ?maxvalue ?p) then (assert (dsp_max (max ?maxi)(2max HFD-P)(3max nil)
                                    (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?b) then (assert (dsp_max (max ?maxi)(2max LFD-B)(3max nil)
                                    (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?bor) then (assert (dsp_max (max ?maxi)(2max border)(3max nil)
                                    (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?ton) then (assert (dsp_max (max ?maxi)(2max tonic)(3max nil)
                                    (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?burst) then (assert (dsp_max (max ?maxi)(2max bursting)(3max nil)
                                    (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?chug) then (assert (dsp_max (max ?maxi)(2max chugging)(3max nil)
                                    (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?paus) then (assert (dsp_max (max ?maxi)(2max pausing)(3max nil)
                                    (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?fib) then (assert (dsp_max (max ?maxi)(2max fiber)(3max nil)
                                    (location_id ?l2)(timetag ?time2)))
else
```

```
 (if (= ?maxvalue ?mul) then (assert (dsp_max (max ?maxi)(2max multiple)(3max nil)
                                              (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?art) then (assert (dsp_max (max ?maxi)(2max artificial)(3max nil)
                                              (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?in) then (assert (dsp_max (max ?maxi)(2max injury)(3max nil)
                                              (location_id ?l2)(timetag ?time2)))
else
 (if (= ?maxvalue ?trem) then (assert (dsp_max (max ?maxi)(2max tremor)(3max nil)
                                              (location_id ?l2)(timetag ?time2)))
))))))))))))))

;CELL ORDER 3 MAX VAL. This rule is responsible for finding the 3rd maximum
;probability and associating the correct cel type with it.
(deffunction cellorder_3maxval (?maxvalue ?pop ?p ?b ?bor ?ton ?burst ?chug ?paus
?fib
                                ?mul ?art ?in ?trem ?maxi ?maxi2 ?l2 ?time2)
 (if (= ?maxvalue ?pop) then (assert (dsp_max (max ?maxi)(2max ?maxi2)
                                              (3max popcorn)
                                              (status complete)(location_id ?l2)
                                              (timetag ?time2)))
else
 (if (= ?maxvalue ?p) then (assert (dsp_max (max ?maxi)(2max ?maxi2)(3max HFD-P)
                                              (status complete)(location_id ?l2)
```

```
                                        (timetag ?time2)))
else
  (if (= ?maxvalue ?b) then (assert (dsp_max (max ?maxi)(2max ?maxi2)(3max LFD-B)
                                        (status complete)(location_id ?l2)
                                        (timetag ?time2)))
else
  (if (= ?maxvalue ?bor) then (assert (dsp_max (max ?maxi)(2max ?maxi2)(3max border)
                                        (status complete)(location_id ?l2)
                                        (timetag ?time2)))
else
  (if (= ?maxvalue ?ton) then (assert (dsp_max (max ?maxi)(2max ?maxi2)(3max tonic)
                                        (status complete)(location_id ?l2)
                                        (timetag ?time2)))
else
  (if (= ?maxvalue ?burst) then (assert (dsp_max (max ?maxi)(2max ?maxi2)
                                        (3max bursting)
                                        (status complete)(location_id ?l2)
                                        (timetag ?time2)))
else
  (if (= ?maxvalue ?chug) then (assert (dsp_max (max ?maxi)(2max ?maxi2)
                                        (3max chugging)
                                        (status complete)(location_id ?l2)
                                        (timetag ?time2)))
else
  (if (= ?maxvalue ?paus) then (assert (dsp_max (max ?maxi)(2max ?maxi2)
```

```
                                        (3max pausing)
                                        (status complete)(location_id ?l2)
                                        (timetag ?time2)))
else
 (if (= ?maxvalue ?fib) then (assert (dsp_max (max ?maxi)(2max ?maxi2)(3max fiber)
                                        (status complete)(location_id ?l2)
                                        (timetag ?time2)))
else
 (if (= ?maxvalue ?mul) then (assert (dsp_max (max ?maxi)(2max ?maxi2)
                                        (3max multiple)
                                        (status complete)(location_id ?l2)
                                        (timetag ?time2)))
else
 (if (= ?maxvalue ?art) then (assert (dsp_max (max ?maxi)(2max ?maxi2)
                                        (3max artificial)
                                        (status complete)(location_id ?l2)
                                        (timetag ?time2)))
else
 (if (= ?maxvalue ?in) then (assert (dsp_max (max ?maxi)(2max ?maxi2)(3max injury)
                                        (status complete)(location_id ?l2)
                                        (timetag ?time2)))
else
 (if (= ?maxvalue ?trem) then (assert (dsp_max (max ?maxi)(2max ?maxi2)
                                        (3max tremor)
                                        (status complete)(location_id ?l2)
```

```
                                        (timetag ?time2)))
)))))))))))))))

;FIND MAX 3 OF PROBABILITIES
;-------------------------------------------------------------------------------------------------------------
;The purpose is to find the maximum 3 cell type probabilities and use them
;as potential cell type identification for a specific depth.  Can be
;any three from the list.
(defrule find_max_cell
        (subgoal name cell status active)
        ?dsp <- (dsp_list (sn_ratio ?sn)(background ?bg)(popcorn ?pop)
                          (HFD-P ?p)(LFD-B ?b)(border ?bor)(tonic ?ton)
                          (bursting ?burst)(chugging ?chug)(pausing ?paus)
                          (fiber ?fib)(multiple ?mul)(artificial ?art)
                          (injury ?in)(tremor ?trem)
                          (location_id ?l1)(timetag ?time1))
        ?order <- (dsp_max (max nil)(2max nil)(3max nil)(status nil)
                           (location_id ?l2)(timetag ?time2))
        (test (eq ?l1 ?l2))
=>
        (retract ?dsp ?order)
        (bind ?maxvalue (find_max ?pop ?p ?b ?bor ?ton ?burst ?chug ?paus
                                  ?fib ?mul ?art ?in ?trem) )
        (bind ?nextcell (replace ?maxvalue ?sn ?bg ?pop ?p ?b ?bor ?ton ?burst ?chug
                                  ?paus ?fib ?mul ?art ?in ?trem ?l1 ?time1) )
```

```
        (bind ?cellorder (cellorder_maxval ?maxvalue ?pop ?p ?b ?bor ?ton
                                           ?burst ?chug ?paus ?fib ?mul
                                           ?art ?in ?trem ?l2 ?time2) )
)
(defrule find_2max_cell
        (subgoal name cell status active)
        ?dsp <- (dsp_list (sn_ratio ?sn)(background ?bg)(popcorn ?pop)
                          (HFD-P ?p)(LFD-B ?b)(border ?bor)(tonic ?ton)
                          (bursting ?burst)(chugging ?chug)(pausing ?paus)
                          (fiber ?fib)(multiple ?mul)(artificial ?art)
                          (injury ?in)(tremor ?trem)
                          (location_id ?l1)(timetag ?time1))
        ?order <- (dsp_max (max ?maxi)(2max nil)(3max nil)(status nil)
                           (location_id ?l2)(timetag ?time2))
        (test (eq ?l1 ?l2))
=>
        (retract ?dsp ?order)
        (bind ?maxvalue (find_max ?pop ?p ?b ?bor ?ton ?burst ?chug ?paus
                                  ?fib ?mul ?art ?in ?trem) )
        (bind ?nextcell (replace ?maxvalue ?sn ?bg ?pop ?p ?b ?bor ?ton ?burst ?chug
                                 ?paus ?fib ?mul ?art ?in ?trem ?l1 ?time1) )
        (bind ?cellorder (cellorder_2maxval ?maxvalue ?pop ?p ?b ?bor ?ton
                                           ?burst ?chug ?paus ?fib ?mul
                                           ?art ?in ?trem ?maxi ?l2 ?time2) )
)
```

```
(defrule find_3max_cell
        (subgoal name cell status active)
        ?dsp <- (dsp_list (sn_ratio ?sn)(background ?bg)(popcorn ?pop)
                          (HFD-P ?p)(LFD-B ?b)(border ?bor)(tonic ?ton)
                          (bursting ?burst)(chugging ?chug)(pausing ?paus)
                          (fiber ?fib)(multiple ?mul)(artificial ?art)
                          (injury ?in)(tremor ?trem)
                          (location_id ?l1)(timetag ?time1))
        ?order <- (dsp_max (max ?maxi)(2max ?maxi2)(3max nil)(status nil)
                           (location_id ?l2)(timetag ?time2))
        (test (eq ?l1 ?l2))
=>
        (retract ?dsp ?order)
        (bind ?maxvalue (find_max ?pop ?p ?b ?bor ?ton ?burst ?chug ?paus
                                  ?fib ?mul ?art ?in ?trem) )
        (bind ?nextcell (replace ?maxvalue ?sn ?bg ?pop ?p ?b ?bor ?ton ?burst ?chug
                                 ?paus ?fib ?mul ?art ?in ?trem ?l1 ?time1) )
        (bind ?cellorder (cellorder_3maxval ?maxvalue ?pop ?p ?b ?bor ?ton
                                            ?burst ?chug ?paus ?fib ?mul
                                            ?art ?in ?trem ?maxi ?maxi2 ?l2
                                            ?time2 ) )
)
```

```
;DECIDING DSP CELL TYPE
;-----------------------------------------------------------------------------------------------------------
;Now comparing the 3 maximum values with the s/n ratio rules we can eliminate some
;answers
(defrule dsp_primary_max1
        (cellgoal name primary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
                (location_id ?l1)(timetag ?time1))
        (cell_layer_combo (layer ST)(cell ?cell1))
        (cell_layer_combo (layer GPe)(cell ?cell2))
        (cell_layer_combo (layer GPi)(cell ?cell3))
        (cell_layer_combo (layer IC)(cell ?cell4))
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)
                (prevhist ?hist)(choice ?cho)(override ?over)
                        (location_id ?l2)(timetag ?time2))
        (test (eq ?l1 ?l2))
        (or (test (eq ?max1 ?cell1))
           (test (eq ?max1 ?cell2))
           (test (eq ?max1 ?cell3))
           (test (eq ?max1 ?cell4)))
=>
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?max1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)(override ?over)

                                        (location_id ?l2)(timetag ?time2)))
```

```
        (assert (done primary max1))
)
(defrule dsp_primary_max2
        (cellgoal name primary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
                (location_id ?l1)(timetag ?time1))
        (cell_layer_combo (layer ST)(cell ?cell1))
        (cell_layer_combo (layer GPe)(cell ?cell2))
        (cell_layer_combo (layer GPi)(cell ?cell3))
        (cell_layer_combo (layer IC)(cell ?cell4))
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)
                        (prevhist ?hist)(choice ?cho)(override ?over)
                        (location_id ?l2)(timetag ?time2))
        (test (eq ?l1 ?l2))
        (or (test (eq ?max2 ?cell1))
           (test (eq ?max2 ?cell2))
           (test (eq ?max2 ?cell3))
           (test (eq ?max2 ?cell4)))
=>
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?max2)(user ?user)
                                (prevhist ?hist)(choice ?cho)(override ?over)
                                (location_id ?l2)(timetag ?time2)))
        (assert (done primary max2))
)
(defrule dsp_primary_max3
```

```
        (cellgoal name primary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
                (location_id ?l1)(timetag ?time1))
        (cell_layer_combo (layer ST)(cell ?cell1))
        (cell_layer_combo (layer GPe)(cell ?cell2))
        (cell_layer_combo (layer GPi)(cell ?cell3))
        (cell_layer_combo (layer IC)(cell ?cell4))
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)
                                (prevhist ?hist)(choice ?cho)(override ?over)
                                    (location_id ?l2)(timetag ?time2))
        (test (eq ?l1 ?l2))
        (or (test (eq ?max3 ?cell1))
           (test (eq ?max3 ?cell2))
           (test (eq ?max3 ?cell3))
           (test (eq ?max3 ?cell4)))
=>
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?max3)(user ?user)
                                        (prevhist ?hist)(choice ?cho)(override ?over)
                                            (location_id ?l2)(timetag ?time2)))
        (assert (done primary max3)))
(defrule dsp_noprimary_max1
        (cellgoal name primary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
                (location_id ?l1)(timetag ?time1))
        (cell_layer_combo (layer any)(cell ?cell))
```

```
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)(prevhist ?hist)
                                (choice ?cho)(override ?over)(location_id ?l2)
                                (timetag ?time2))
        (test (eq ?l1 ?l2))
        (or (test (eq ?max1 ?cell))
           (test (eq ?max1 fiber)))
        (test (neq ?max1 artificial))
=>
        (assert (done primary max1))
        (assert (deciding_cellid_layer (anat ?anat)(dsp none)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                           (override ?over)(location_id ?l2)
                                           (timetag ?time2)))
)
(defrule dsp_specprimary_max1
        (cellgoal name primary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
                 (location_id ?l1)(timetag ?time1))
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)(prevhist ?hist)
                                (choice ?cho)(override ?over)(location_id ?l2))
        (control (name current)(location_id ?l3)(depth ?d)(time ?time3))
        (test (eq ?l1 ?l2 ?l3))
        (test (eq ?max1 artificial))
=>
        (assert (done primary max1))
```

```
        (assert (goal interface_initiated status active))
        (assert (output (type M)(format message)(variable nil)
                        (text-field DSP is picking up a lot of artificial sound.
                         Please chek equipment.)(depth ?d)(timetag ?time3)))
        (assert (deciding_cellid_layer (anat ?anat)(dsp none)(user ?user)
                                       (prevhist ?hist)(choice ?cho)
                                          (override ?over)(location_id ?l2)
                                          (timetag ?time3)))
)
(defrule dsp_noprimary_max2
        (cellgoal name primary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
                 (location_id ?l1)(timetag ?time1))
        (cell_layer_combo (layer any)(cell ?cell))
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)(prevhist ?hist)
                               (choice ?cho)(override ?over)(location_id ?l2)
                                 (timetag ?time2))
        (test (eq ?l1 ?l2))
        (or (test (eq ?max2 ?cell))
           (test (eq ?max2 fiber)))
=>
        (assert (done primary max2))
        (assert (deciding_cellid_layer (anat ?anat)(dsp none)(user ?user)
                                       (prevhist ?hist)(choice ?cho)
                                          (override ?over)(location_id ?l2)
```

```
                                        (timetag ?time2)))
)
(defrule dsp_noprimary_max3
        (cellgoal name primary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
                (location_id ?l1)(timetag ?time1))
        (cell_layer_combo (layer any)(cell ?cell))
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)(prevhist ?hist)
                                (choice ?cho)(override ?over)(location_id ?l2)
                                (timetag ?time2))
        (test (eq ?l1 ?l2))
        (or (test (eq ?max3 ?cell))
           (test (eq ?max3 fiber)))
=>
        (assert (done primary max3))
        (assert (deciding_cellid_layer (anat ?anat)(dsp none)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                           (override ?over)(location_id ?l2)
                                           (timetag ?time2)))
)
(defrule dsp_primary_done
        ?goal <- (cellgoal name primary status active)
        ?max1 <- (done primary max1)
        ?max2 <- (done primary max2)
        ?max3 <- (done primary max3)
```

```
        ?decide <- (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)
                                          (prevhist ?hist)(choice ?cho)
                                          (override ?over)(location_id ?l1)
                                          (timetag ?time1))
=>
        (retract ?goal ?max1 ?max2 ?max3 ?decide)
        (assert (cellgoal name assign status complete))
)
(defrule dsp_secondary_max1
        (cellgoal name secondary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
                 (location_id ?l1)(timetag ?time1))
        (cell_layer_combo (layer Le)(cell ?cell1))
        (cell_layer_combo (layer Li)(cell ?cell2))
        (cell_layer_combo (layer La)(cell ?cell3))
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)
                               (prevhist ?hist)(choice ?cho)
                               (override ?over)(location_id ?l2)
                               (timetag ?time2))
        (test (eq ?l1 ?l2))
        (or (test (eq ?max1 ?cell1))
           (test (eq ?max1 ?cell2))
           (test (eq ?max1 ?cell3)))
=>
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?max1)(user ?user)
```

```
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l2)
                                        (timetag ?time2)))
        (assert (done secondary max1))
)
(defrule dsp_secondary_max2
        (cellgoal name secondary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
                (location_id ?l1)(timetag ?time1))
        (cell_layer_combo (layer Le)(cell ?cell1))
        (cell_layer_combo (layer Li)(cell ?cell2))
        (cell_layer_combo (layer La)(cell ?cell3))
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)
                                (prevhist ?hist)(choice ?cho)
                                (override ?over)(location_id ?l2)
                                (timetag ?time2))
        (test (eq ?l1 ?l2))
        (or (test (eq ?max2 ?cell1))
           (test (eq ?max2 ?cell2))
           (test (eq ?max2 ?cell3)))
=>
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?max2)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l2)
                                        (timetag ?time2)))
```

```
        (assert (done secondary max2))
)
(defrule dsp_secondary_max3
        (cellgoal name secondary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
                (location_id ?l1)(timetag ?time1))
        (cell_layer_combo (layer Le)(cell ?cell1))
        (cell_layer_combo (layer Li)(cell ?cell2))
        (cell_layer_combo (layer La)(cell ?cell3))
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)
                                (prevhist ?hist)(choice ?cho)
                                (override ?over)(location_id ?l2)
                                (timetag ?time2))
        (test (eq ?l1 ?l2))
        (or (test (eq ?max3 ?cell1))
           (test (eq ?max3 ?cell2))
           (test (eq ?max3 ?cell3)))
=>
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?max3)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l2)
                                        (timetag ?time2)))
        (assert (done secondary max3))
)
```

```
(defrule dsp_nosecondary_max1
        (cellgoal name secondary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
                (location_id ?l1)(timetag ?time1))
        (cell_layer_combo (layer ST)(cell ?cell1))
        (cell_layer_combo (layer GPe)(cell ?cell2))
        (cell_layer_combo (layer GPi)(cell ?cell3))
        (cell_layer_combo (layer any)(cell ?cell4))
        (cell_layer_combo (layer IC)(cell ?cell5))
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)(prevhist ?hist)
                                (choice ?cho)(override ?over)(location_id ?l2)
                                (timetag ?time2))
        (test (eq ?l1 ?l2))
        (or (test (eq ?max1 ?cell1))
           (test (eq ?max1 ?cell2))
           (test (eq ?max1 ?cell3))
           (test (eq ?max1 ?cell4))
           (test (eq ?max1 ?cell5)))
        (test (neq ?max1 border))
        (test (neq ?max1 artificial))
=>
        (assert (done secondary max1))
        (assert (deciding_cellid_layer (anat ?anat)(dsp none)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                           (override ?over)(location_id ?l2)
```

```
                                        (timetag ?time2)))      )
(defrule dsp_specsecondary_max1
        (cellgoal name secondary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
                (location_id ?l1)(timetag ?time1))
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)(prevhist ?hist)
                                (choice ?cho)(override ?over)(location_id ?l2)
                                (timetag ?time2))
        (control (name current)(location_id ?l3)(depth ?d)(time ?time3))
        (test (eq ?l1 ?l2 ?l3))
        (test (eq ?max1 artificial))
=>
        (assert (done secondary max1))
        (assert (goal interface_initiated status active))
        (assert (output (type M)(format message)(variable nil)
                        (text-field DSP is picking up a lot of artificial sound.
                         Please check equipment.)(depth ?d)(timetag ?time3)))
        (assert (deciding_cellid_layer (anat ?anat)(dsp none)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l2)
                                        (timetag ?time2)))
)
(defrule dsp_nosecondary_max2
        (cellgoal name secondary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
```

```
                (location_id ?l1)(timetag ?time1))
        (cell_layer_combo (layer ST)(cell ?cell1))
        (cell_layer_combo (layer GPe)(cell ?cell2))
        (cell_layer_combo (layer GPi)(cell ?cell3))
        (cell_layer_combo (layer any)(cell ?cell4))
        (cell_layer_combo (layer IC)(cell ?cell5))
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)(prevhist ?hist)
                               (choice ?cho)(override ?over)(location_id ?l2)
                               (timetag ?time2))
        (test (eq ?l1 ?l2))
        (or (test (eq ?max2 ?cell1))
           (test (eq ?max2 ?cell2))
           (test (eq ?max2 ?cell3))
           (test (eq ?max2 ?cell4))
           (test (eq ?max2 ?cell5)))
=>
        (assert (done secondary max2))
        (assert (deciding_cellid_layer (anat ?anat)(dsp none)(user ?user)
                                       (prevhist ?hist)(choice ?cho)
                                          (override ?over)(location_id ?l2)
                                          (timetag ?time2)))
)
(defrule dsp_nosecondary_max3
        (cellgoal name secondary status active)
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(status complete)
```

```
                (location_id ?l1)(timetag ?time1))
        (cell_layer_combo (layer ST)(cell ?cell1))
        (cell_layer_combo (layer GPe)(cell ?cell2))
        (cell_layer_combo (layer GPi)(cell ?cell3))
        (cell_layer_combo (layer any)(cell ?cell4))
        (cell_layer_combo (layer IC)(cell ?cell5))
        (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)(prevhist ?hist)
                               (choice ?cho)(override ?over)(location_id ?l2)
                               (timetag ?time2))
        (test (eq ?l1 ?l2))
        (or (test (eq ?max3 ?cell1))
           (test (eq ?max3 ?cell2))
           (test (eq ?max3 ?cell3))
           (test (eq ?max3 ?cell4))
           (test (eq ?max3 ?cell5)))
=>
        (assert (done secondary max3))
        (assert (deciding_cellid_layer (anat ?anat)(dsp none)(user ?user)
                                       (prevhist ?hist)(choice ?cho)
                                          (override ?over)(location_id ?l2)
                                          (timetag ?time2)))
)
(defrule dsp_secondary_done
        ?goal <- (cellgoal name secondary status active)
        ?max1 <- (done secondary max1)
```

```
        ?max2 <- (done secondary max2)
        ?max3 <- (done secondary max3)
        ?decide <- (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)
                                          (prevhist ?hist)(choice ?cho)
                                          (override ?over)(location_id ?l1)
                                          (timetag ?time1))
=>
        (retract ?goal ?max1 ?max2 ?max3 ?decide)
        (assert (cellgoal name assign status complete))
)

;NO CELL TYPE
;-----------------------------------------------------------------------------------------------------------
;Whenever DSP and User cell type selection is not available use this rule to get through
;the mechanism of cell type decisions.
(defrule no_dsp
        (subgoal name cell status active)
        ?decide <- (deciding_cellid_layer (anat ?anat)(dsp nil)(user ?user)
                                          (prevhist ?hist)(choice ?cho)
                                          (override ?over)(location_id ?l1)
                                          (timetag ?time1))
        ?dsp <- (no dsp input)
=>
        (retract ?decide ?dsp)
        (assert (deciding_cellid_layer (anat ?anat)(dsp none)(user ?user)
```

```
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
        (assert (cellgoal name assign status complete))
)
```

D.12. Cell Anatomy

```
;       ---------------------------------------------------------------------------------------------------
;                       Cell Anatomy
;       Program:        Pallidtomy Version 6.0
;       Author:         Linda Harley
;       Date:           September 11, 2003
;       File:           cellanat.clp
;       Updates:        Oct 8, 2003     Added "cell_expected_none" rule.
;       ---------------------------------------------------------------------------------------------------
;CELL ANATOMY
;---------------------------------------------------------------------------------------------------------
;From comparing current depth to pathplan we determine the expected layer.
;Knowledge exist for what cell types compare to what layers, so we kind of
;know what cell types to expect, or at least to be looking for.
(defrule cell_expected_done
        (subgoal name depth status active)
        (not (deciding_cellid_layer (anat nil)(dsp ?dsp)(user ?user)
                                    (prevhist ?hist)(choice ?cho)
                                    (override ?over)(location_id ?l1)
                                    (timetag ?time1)))
=>
        (assert (depthsubgoal name cellanat status complete))
)
(defrule cell_expected_none
        (subgoal name depth status active)
```

```
        ?cellid <- (deciding_cellid_layer (anat nil)(dsp ?dsp1)(user ?user1)
                                          (prevhist ?hist)(choice ?cho)
                                          (override ?over)(location_id ?l1)
                                          (timetag ?time1))
        (deciding_layer (depth none)(cell ?dsp2)(user ?user2)
                        (decision ?dec)(location_id ?l2)(timetag ?time2))
        (cell_layer_combo (layer none)(cell ?cell))
        (test (eq ?l1 ?l2))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat none)(dsp ?dsp1)(user ?user1)
                                       (prevhist ?hist)(choice ?cho)
                                       (override ?over)(location_id ?l1)
                                       (timetag ?time1)))
)
(defrule cell_expected_ST
        (subgoal name depth status active)
        ?cellid <- (deciding_cellid_layer (anat nil)(dsp ?dsp1)(user ?user)
                                          (prevhist ?hist)(choice ?cho)
                                          (override ?over)(location_id ?l1)
                                          (timetag ?time1))
        (deciding_layer (depth ST)(cell ?dsp2)(user ?user2)(decision ?dec)
                        (location_id ?l2)(timetag ?time2))
        (cell_layer_combo (layer ST)(cell ?cell))
        (test (eq ?l1 ?l2))
```

```
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat popcorn)(dsp ?dsp1)(user ?user)
                                       (prevhist ?hist)(choice ?cho)
                                       (override ?over)(location_id ?l1)
                                       (timetag ?time1)))
        (assert (deciding_cellid_layer (anat border)(dsp ?dsp1)(user ?user)
                                       (prevhist ?hist)(choice ?cho)
                                       (override ?over)(location_id ?l1)
                                       (timetag ?time1)))
        (assert (deciding_cellid_layer (anat caudate)(dsp ?dsp1)(user ?user)
                                       (prevhist ?hist)(choice ?cho)
                                       (override ?over)(location_id ?l1)
                                       (timetag ?time1)))
)
(defrule cell_expected_Le
        (subgoal name depth status active)
        ?cellid <- (deciding_cellid_layer (anat nil)(dsp ?dsp1)(user ?user)
                                          (prevhist ?hist)(choice ?cho)
                                          (override ?over)(location_id ?l1)
                                          (timetag ?time1))
        (deciding_layer (depth Le)(cell ?dsp2)(user ?user2)(decision ?dec)
                        (location_id ?l2)(timetag ?time2))
        (cell_layer_combo (layer Le)(cell ?cell))
        (test (eq ?l1 ?l2))
```

```
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat fiber)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
        (assert (deciding_cellid_layer (anat border)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
)
(defrule cell_expected_GPe
        (subgoal name depth status active)
        ?cellid <- (deciding_cellid_layer (anat nil)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1))
        (deciding_layer (depth GPe)(cell ?dsp2)(user ?user2)(decision ?dec)
                        (location_id ?l2)(timetag ?time2))
        (cell_layer_combo (layer GPe)(cell ?cell))
        (test (eq ?l1 ?l2))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat HFD-P)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
```

```
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
        (assert (deciding_cellid_layer (anat LFD-B)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
        (assert (deciding_cellid_layer (anat border)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
)
(defrule cell_expected_Li
        (subgoal name depth status active)
        ?cellid <- (deciding_cellid_layer (anat nil)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1))
        (deciding_layer (depth Li)(cell ?dsp2)(user ?user2)(decision ?dec)
                        (location_id ?l2)(timetag ?time2))
        (cell_layer_combo (layer Li)(cell ?cell))
        (test (eq ?l1 ?l2))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat fiber)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
```

```
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
        (assert (deciding_cellid_layer (anat border)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
)
(defrule cell_expected_GPi
        (subgoal name depth status active)
        ?cellid <- (deciding_cellid_layer (anat nil)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1))
        (deciding_layer (depth GPi)(cell ?dsp2)(user ?user2)(decision ?dec)
                        (location_id ?l2)(timetag ?time2))
        (cell_layer_combo (layer GPi)(cell ?cell))
        (test (eq ?l1 ?l2))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat tonic)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
        (assert (deciding_cellid_layer (anat bursting)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
```

```
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
        (assert (deciding_cellid_layer (anat chugging)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
        (assert (deciding_cellid_layer (anat pausing)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
        (assert (deciding_cellid_layer (anat tremor)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
        (assert (deciding_cellid_layer (anat border)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1)))
)
(defrule cell_expected_La
        (subgoal name depth status active)
        ?cellid <- (deciding_cellid_layer (anat nil)(dsp ?dsp1)(user ?user)
                                        (prevhist ?hist)(choice ?cho)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1))
```

```
        (deciding_layer (depth La)(cell ?dsp2)(user ?user2)(decision ?dec)
                        (location_id ?l2)(timetag ?time2))
        (cell_layer_combo (layer La)(cell ?cell))
        (test (eq ?l1 ?l2))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat fiber)(dsp ?dsp1)(user ?user)
                                       (prevhist ?hist)(choice ?cho)
                                       (override ?over)(location_id ?l1)
                                       (timetag ?time1)))
        (assert (deciding_cellid_layer (anat border)(dsp ?dsp1)(user ?user)
                                       (prevhist ?hist)(choice ?cho)
                                       (override ?over)(location_id ?l1)
                                       (timetag ?time1)))
)
(defrule cell_expected_IC
        (subgoal name depth status active)
        ?cellid <- (deciding_cellid_layer (anat nil)(dsp ?dsp1)(user ?user)
                                          (prevhist ?hist)(choice ?cho)
                                          (override ?over)(location_id ?l1)
                                          (timetag ?time1))
        (deciding_layer (depth IC)(cell ?dsp2)(user ?user2)(decision ?dec)
                        (location_id ?l2)(timetag ?time2))
        (cell_layer_combo (layer IC)(cell ?cell))
        (test (eq ?l1 ?l2))
```

```
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat multiple)(dsp ?dsp1)(user ?user)
                                       (prevhist ?hist)(choice ?cho)
                                       (override ?over)(location_id ?l1)
                                       (timetag ?time1)))
        (assert (deciding_cellid_layer (anat injury)(dsp ?dsp1)(user ?user)
                                       (prevhist ?hist)(choice ?cho)
                                       (override ?over)(location_id ?l1)
                                       (timetag ?time1)))
        (assert (deciding_cellid_layer (anat neg)(dsp ?dsp1)(user ?user)
                                       (prevhist ?hist)(choice ?cho)
                                       (override ?over)(location_id ?l1)
                                       (timetag ?time1)))
)
(defrule cell_expected_OT
        (subgoal name depth status active)
        ?cellid <- (deciding_cellid_layer (anat nil)(dsp ?dsp1)(user ?user)
                                          (prevhist ?hist)(choice ?cho)
                                          (override ?over)(location_id ?l1)
                                          (timetag ?time1))
        (deciding_layer (depth OT)(cell ?dsp2)(user ?user2)(decision ?dec)
                        (location_id ?l2)(timetag ?time2))
        (cell_layer_combo (layer OT)(cell ?cell))
        (test (eq ?l1 ?l2))
```

```
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat ?cell)(dsp ?dsp1)(user ?user)
                                       (prevhist ?hist)(choice ?cho)
                                       (override ?over)(location_id ?l1)
                                       (timetag ?time1)))
)
```

D.13. User

```
;       ---------------------------------------------------------------------------------------------------
;                       User
;       Program:        Pallidotomy Version 6.0
;       Author:         Dr. N Baker and Linda Harley
;       Date:           September 1, 2003
;       File:           user.clp
;       Updates:        Sep 1           Starting reconstruction of KBS.
;                       Sep 12          Updated user_cell, user_cell_done and
;                                       user_cell_none
;                       Oct 8           Added "user_layer_none"
;       ---------------------------------------------------------------------------------------------------
;This file contains the process knowledge that makes up part of the user
;direct interface with the KBS within the Onetrack forward chaining process.
(defrule user_motor
        (subgoal name user status active)
=>
        (assert (usergoal name user_motor status complete) )
)
(defrule user_cell
        (subgoal name user status active)
        (user (layer ?lay)(cell ?cell)(location_id ?l1)(timetag ?time1))
        ?cellid <- (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user nil)
                                          (location_id ?l2)(timetag ?time2))
        (test (eq ?l1 ?l2))
```

```
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?cell)
                                        (location_id ?l2)(timetag ?time2)))
)
(defrule user_cell_done
        (subgoal name user status active)
        (not (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user nil)
                                    (location_id ?l2)(timetag ?time2)))
=>
        (assert (usergoal name user_cell status complete))
)
(defrule user_cell_none
        (subgoal name user status active)
        ?cellid <- (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user nil)
                                        (location_id ?l1)(timetag ?time2))
        (no user input)
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user none)
                                    (location_id ?l1)(timetag ?time2)))
)
(defrule user_layer
        (subgoal name user status active)
        (user (layer ?lay)(cell ?cell)(location_id ?l1)(timetag ?time1))
```

```
        ?layerid <- (deciding_layer (depth ?dep)(cell ?dsp)(user nil)
                                    (decision ?dec)(location_id ?l2)
                                    (timetag ?time2))
        (test (eq ?l1 ?l2))
=>
        (retract ?layerid)
        (assert (deciding_layer (depth ?dep)(cell ?dsp)(user ?lay)
                                (decision ?dec)(location_id ?l2)
                                (timetag ?time2)))
        (assert (usergoal name user_layer status complete) )
)
(defrule user_layer_none
        (subgoal name user status active)
        ?user <- (no user input)
        ?layerid <- (deciding_layer (depth ?dep)(cell ?dsp)(user nil)
                                    (decision ?dec)(location_id ?l2)
                                    (timetag ?time2))
=>
        (retract ?layerid ?user)
        (assert (deciding_layer (depth ?dep)(cell ?dsp)(user none)
                                (decision ?dec)(location_id ?l2)
                                (timetag ?time2)))
        (assert (usergoal name user_layer status complete))
)
```

```
(defrule user_done_input
        (subgoal name user status active)
        ?motor <- (usergoal name user_motor status complete)
        ?cell <- (usergoal name user_cell status complete)
        ?layer <- (usergoal name user_layer status complete)
=>
        (retract ?motor)
        (retract ?cell)
        (retract ?layer)
        (assert (usergoal name user_input status complete) )
)
```

D.14. Compare Cell Type

```
;       ------------------------------------------------------------------------------------------------------
;                       Compare Cell Type
;       Program:        Pallidotomy Versrion 6.0
;       Author:         Linda Harley
;       Date:           September 12, 2003
;       File:           comparecell.clp
;       Updates:        Oct 8, 2003     Added "compare_cell_cunod_cumatch" and
;                                       "compare_cell_cunod_nomatch"
;       ------------------------------------------------------------------------------------------------------
;The following rules all relate to comparing the cell type selections done by
;anatomical facts, dsp probabilities and user inputs.
(defrule compare_cell_cunoa_cumatch
        (subgoal name infer_layer status active)
        ?cellid <- (deciding_cellid_layer (anat none)(dsp ?dsp)(user ?user)
                                          (prevhist ?hist)(choice nil)
                                          (override ?over)(location_id ?l1)
                                          (timetag ?time1))
        (test (eq ?dsp ?user))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat none)(dsp ?dsp)(user ?user)
                                       (prevhist ?hist)(choice ?user)
                                       (override none)(location_id ?l1)
                                       (timetag ?time1)))
```

```
        (assert (inferlayergoal name infercell status complete))
)
(defrule compare_cell_ainput_nomatch
        (subgoal name infer_layer status active)
        ?cellid <- (deciding_cellid_layer (anat ?anat)(dsp none)(user none)
                                          (prevhist ?hist)(choice nil)
                                          (override ?over)(location_id ?l1)
                                          (timetag ?time1))
        (test (neq ?anat none))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat ?anat)(dsp none)(user none)
                                         (prevhist ?hist)(choice none)
                                         (override none)(location_id ?l1)
                                         (timetag ?time1)))
        (assert (inferlayergoal name infercell status complete))
)
(defrule compare_cell_cunod_nomatch
        (subgoal name infer_layer status active)
        ?cellid <- (deciding_cellid_layer (anat none)(dsp ?dsp)(user ?user)
                                          (prevhist ?hist)(choice nil)
                                          (override ?over)(location_id ?l1)
                                          (timetag ?time1))
        (test (neq ?dsp ?user))
=>
```

```
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat none)(dsp ?dsp)(user ?user)
                                       (prevhist ?hist)(choice ?user)
                                       (override none)(location_id ?l1)
                                       (timetag ?time1)))
        (assert (inferlayergoal name infercell status complete))
)
(defrule compare_cell_allinput_allmatch
        (subgoal name infer_layer status active)
        ?cellid <- (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user)
                                       (prevhist ?hist)(choice nil)
                                       (override ?over)(location_id ?l1)
                                       (timetag ?time1))
        (test (eq ?anat ?dsp ?user))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user)
                                       (prevhist ?hist)(choice ?user)
                                       (override none)(location_id ?l1)
                                       (timetag ?time1)))
        (assert (inferlayergoal name infercell status complete))
)
(defrule compare_cell_allinput_admatch
        (subgoal name infer_layer status active)
        ?cellid <- (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user)
```

```
                                        (prevhist ?hist)(choice nil)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1))
        (test (eq ?anat ?dsp))
        (test (neq ?anat ?user))
        (test (neq ?dsp ?user))
        (test (neq ?user none))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user)
                                        (prevhist ?hist)(choice nil)
                                        (override yes)(location_id ?l1)
                                        (timetag ?time1)))
        (assert (goal interface_initiated status active))
        (assert (output (type Q)(format MC)(variable popcorn HFD-P LFD-B border
                        tonic bursting chugging pausing fiber multiple
                        artificial injury tremor)(text-field Users celtype
                        selection conflicts with pathplan and dsp analysis.
                        Please reselect celltype.)))
)
(defrule compare_cell_verify
        (subgoal name infer_layer status active)
        ?answer <- (answer_received (multichoice ?mc)(timetag ?time1))
        ?cellid <- (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user)
                                        (prevhist ?hist)(choice nil)
```

```
                                        (override yes)(location_id ?l1)
                                        (timetag ?time2))
        (test (> ?time1 ?time2))
=>
        (retract ?answer ?cellid)
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user)
                                        (prevhist ?hist)(choice ?mc)
                                        (override none)(location_id ?l1)
                                        (timetag ?time2)))
        (assert (inferlayergoal name infercell status complete))
)
(defrule compare_cell_allinput_aumatch
        (subgoal name infer_layer status active)
        ?cellid <- (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user)
                                        (prevhist ?hist)(choice nil)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1))
        (test (eq ?anat ?user))
        (test (neq ?anat ?dsp))
        (test (neq ?user ?dsp))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user)
                                        (prevhist ?hist)(choice ?user)
                                        (override none)(location_id ?l1)
```

```
                                        (timetag ?time1)))
        (assert (inferlayergoal name infercell status complete))
)
(defrule compare_cell_allinput_dumatch
        (subgoal name infer_layer status active)
        ?cellid <- (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user)
                                        (prevhist ?hist)(choice nil)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1))
        (test (eq ?dsp ?user))
        (test (neq ?dsp ?anat))
        (test (neq ?user ?anat))
        (test (neq ?anat none))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user)
                                        (prevhist ?hist)(choice ?user)
                                        (override none)(location_id ?l1)
                                        (timetag ?time1)))
        (assert (inferlayergoal name infercell status complete))
)
(defrule compare_cell_allinput_nomatch
        (subgoal name infer_layer status active)
        ?cellid <- (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user)
                                        (prevhist ?hist)(choice nil)
```

```
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1))
        (test (neq ?anat ?dsp ?user))
        (test (neq ?anat none))
        (test (and (neq ?dsp none)
                   (neq ?user none)))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user)
                                       (prevhist ?hist)(choice ?user)
                                       (override none)(location_id ?l1)
                                       (timetag ?time1)))
        (assert (inferlayergoal name infercell status complete))
)
(defrule compare_cell_adinput_allmatch
        (subgoal name infer_layer status active)
        ?cellid <- (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user none)
                                        (prevhist ?hist)(choice nil)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1))
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(location_id ?l2)(timetag
?time2))
        (test (eq ?l1 ?l2))
        (test (eq ?max1 ?dsp ?anat))
=>
```

```
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user none)
                                       (prevhist ?hist)(choice ?dsp)
                                       (override none)(location_id ?l1)
                                       (timetag ?time1)))
        (assert (inferlayergoal name infercell status complete))
)

(defrule compare_cell_adinput_halfmatch
        (subgoal name infer_layer status active)
        ?cellid <- (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user none)
                                        (prevhist ?hist)(choice nil)
                                        (override ?over)(location_id ?l1)
                                        (timetag ?time1))
        (dsp_max (max ?max1)(2max ?max2)(3max ?max3)(location_id ?l2)(timetag
?time2))
        (test (eq ?l1 ?l2))
        (test (neq ?dsp ?max1))
        (test (eq ?dsp ?anat))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user none)
                                       (prevhist ?hist)(choice none)
                                       (override none)(location_id ?l1)
                                       (timetag ?time1)))
```

```
        (assert (inferlayergoal name infercell status complete))
)
(defrule compare_cell_adinput_nomatch
        (subgoal name infer_layer status active)
        ?cellid <- (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user none)
                                          (prevhist ?hist)(choice nil)
                                          (override ?over)(location_id ?l1)
                                          (timetag ?time1))
        (test (neq ?anat ?dsp))
        (test (and (neq ?anat none)
                   (neq ?dsp none)))
=>
        (retract ?cellid)
        (assert (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user none)
                                       (prevhist ?hist)(choice none)
                                       (override none)(location_id ?l1)

                                       (timetag ?time1)))
        (assert (inferlayergoal name infercell status complete))
)
(defrule compare_cell_cleanup
        (inferlayergoal name infercell status complete)
        (deciding_cellid_layer (anat ?anat1)(dsp ?dsp1)(user ?user1)
                               (prevhist ?hist1)(choice ?cho1)(override none)
                               (location_id ?l1)(timetag ?time1))
```

```
        ?cellid <- (deciding_cellid_layer (anat ?anat2)(dsp ?dsp2)(user ?user2)
                                          (prevhist ?hist2)(choice nil)
                                          (override nil)(location_id ?l2)
                                          (timetag ?time2))
        (test (eq ?l1 ?l2))
=>
        (retract ?cellid)
)
(defrule compare_cell_done
        (inferlayergoal name infercell status complete)
        (not (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user)
                                    (prevhist ?hist)(choice nil)(override nil)
                                    (location_id ?l1)(timetag ?time2)))
=>
        (assert (inferlayergoal name cellcleanup status complete))
)

;The CELL TO LAYER rule converts the final chosen cell type to the correct
;layer type associated with that cell.
(defrule cell_to_layer
        (subgoal name infer_layer status active)
        (deciding_cellid_layer (anat ?anat)(dsp ?dsp)(user ?user1)
                               (prevhist ?hist)(choice ?cho)(override none)
                               (location_id ?l1)(timetag ?time1))
        ?layerid <- (deciding_layer (depth ?depth)(cell nil)(user ?user2)
```

```
                                (decision ?dec)(location_id ?l2)
                                (timetag ?time2))
        ?controller <- (control (name current)(location_id ?l3)(previouse_id ?p3)
                                (track_id ?t3)(depth ?d3)(time ?time3)(layer nil)
                                (cell nil))
        (cell_layer_combo (layer ?layer)(cell ?cell))
        (test (eq ?l1 ?l2 ?l3))
        (test (eq ?cell ?cho))
=>
        (retract ?layerid ?controller)
        (assert (deciding_layer (depth ?depth)(cell ?layer)(user ?user2)
                                (decision ?dec)(location_id ?l2)(timetag ?time2)))
        (assert (inferlayergoal name celltolayer status complete))
        (assert (control (name current)(location_id ?l3)(previouse_id ?p3)
                        (track_id ?t3)(depth ?d3)(time ?time3)(layer nil)
                        (cell ?cho)))
)
```

D.15. Infer Layer

```
;       ---------------------------------------------------------------------------------------------------
;                       Infer Layer
;       Program:        Pallidotomy Version 6.0
;       Author:         Linda Harley and Dr. N Baker
;       Date:           Septeber 4, 2003
;       File:           inferlayer.clp
;       Updates:        Sep 4           Starting reconstruction of KBS.
;                                       Also included the new pathplan update rules
;       ---------------------------------------------------------------------------------------------------


;DETERMINE LAYER
;---------------------------------------------------------------------------------------------------
;The following rules compare all the layer determinations by the different
;phases and comes up with one single conclusion of what layer the probe tip
;is in.  First case is real easy, only depth reasoning.
(defrule determine_layer_dcusame
        (subgoal name infer_layer status active)
        ?decide <- (deciding_layer (depth ?dep)(cell ?cell)(user ?user)
                                   (decision nil)(location_id ?l1)
                                   (timetag ?time1))
        ?controler <- (control (name current)(location_id ?l2)(previouse_id ?p2)
                               (track_id ?t2)(depth ?d)(time ?time2)(layer nil)
                               (cell ?cell2))
        (test (eq ?l1 ?l2))
```

```
        (test (eq ?dep ?cell ?user))
=>
        (retract ?decide ?controler)
        (assert (deciding_layer (depth ?dep)(cell ?cell)(user ?user)
                                (decision ?user)(location_id ?l1)
                                (timetag ?time1)))
        (assert (control (name current)(location_id ?l2)(previouse_id ?p2)
                        (track_id ?t2)(depth ?d)(time ?time2)(layer ?user)
                        (cell ?cell2)))
        (assert (inferlayergoal name determine_layer status complete))
)
(defrule determine_layer_udsame
        (subgoal name infer_layer status active)
        ?decide <- (deciding_layer (depth ?dep)(cell ?cell)(user ?user)
                                   (decision nil)(location_id ?l1)
                                   (timetag ?time1))
        ?controler <- (control (name current)(location_id ?l2)(previouse_id ?p2)
                                (track_id ?t2)(depth ?d)(time ?time2)(layer nil)
                                (cell ?cell2))
        (test (eq ?l1 ?l2))
        (test (eq ?user ?dep))
        (test (neq ?user ?cell))
        (test (neq ?cell nil))
=>
        (retract ?decide ?controler)
```

```
        (assert (deciding_layer (depth ?dep)(cell ?cell)(user ?user)
                                (decision ?user)(location_id ?l1)
                                (timetag ?time1)))
        (assert (control (name current)(location_id ?l2)(previouse_id ?p2)
                        (track_id ?t2)(depth ?d)(time ?time2)(layer ?user)
                        (cell ?cell2)))
        (assert (inferlayergoal name determine_layer status complete))
)
(defrule determine_layer_ucsame
        (subgoal name infer_layer status active)
        ?decide <- (deciding_layer (depth ?dep)(cell ?cell)(user ?user)
                                   (decision nil)(location_id ?l1)(timetag ?time1))
        ?controler <- (control (name current)(location_id ?l2)(previouse_id ?p2)
                                (track_id ?t2)(depth ?d)(time ?time2)(layer nil)
                                (cell ?cell2))
        (test (eq ?l1 ?l2))
        (test (eq ?user ?cell))
        (test (neq ?user ?dep))
        (test (and (neq ?cell none)
                   (neq ?user none)))
=>
        (retract ?decide ?controler)
        (assert (deciding_layer (depth ?dep)(cell ?cell)(user ?user)
                                (decision ?user)(location_id ?l1)
                                (timetag ?time1)))
```

```
        (assert (control (name current)(location_id ?l2)(previouse_id ?p2)
                         (track_id ?t2)(depth ?d)(time ?time2)(layer ?user)
                         (cell ?cell2)))
        (assert (inferlayergoal name determine_layer status complete))
)
(defrule determine_layer_cdsame
        (subgoal name infer_layer status active)
        ?decide <- (deciding_layer (depth ?dep)(cell ?cell)(user ?user)
                                   (decision nil)(location_id ?l1)(timetag ?time1))
        ?controler <- (control (name current)(location_id ?l2)(previouse_id ?p2)
                               (track_id ?t1)(depth ?d)(time ?time2)(layer nil)
                               (cell ?cell2))
        (test (eq ?l1 ?l2))
        (test (eq ?dep ?cell))
        (test (neq ?dep ?user))
        (test (neq ?user none))
=>
        (retract ?decide)
        (assert (deciding_layer (depth ?dep)(cell ?cell)(user ?user)
                                (decision askuser)(location_id ?l1)
                                (timetag ?time1)))
        (assert (goal interface_initiated status active))
        (assert (output (type Q)(format MC)(variable Striatum Le GPe Li GPi La
                        Optic InternalCapsule)(text-field "According to KBS the probe
                        is in layer" ?dep  "User is in conflict.  Please re-select
```

```
                    layer type.")(depth ?d)(timetag ?time1)))
)
(defrule determine_layer_verify
        (subgoal name infer_layer status active)
        ?answer <- (answer_received (multichoice ?mc)(timetag ?time1))
        ?decide <- (deciding_layer (depth ?dep)(cell ?cell)(user ?user)
                                   (decision askuser)(location_id ?l2)
                                   (timetag ?time2))
        ?control <- (control (name current)(location_id ?l3)(previouse_id ?p3)
                             (track_id ?t3)(depth ?d)(time ?time3)(layer nil)
                             (cell ?cell3))
        (test (eq ?l2 ?l3))
        (test (> ?time1 ?time2))
=>
        (retract ?answer ?decide ?control)
        (assert (deciding_layer (depth ?dep)(cell ?cell)(user ?user)
                                (decision ?mc)(location_id ?l2)(timetag ?time2)))
        (assert (control (name current)(location_id ?l3)(previouse_id ?p3)
                         (track_id ?t3)(depth ?d)(time ?time3)(layer ?mc)
                         (cell ?cell3)))
        (assert (inferlayergoal name determine_layer status complete))
)
(defrule determine_layer_dcunotsame
        (subgoal name infer_layer status active)
        ?decide <- (deciding_layer (depth ?dep)(cell ?cell)(user ?user)
```

```
                                (decision nil)(location_id ?l1)(timetag ?time1))
        ?control <- (control (name current)(location_id ?l2)(previouse_id ?p2)
                (track_id ?t1)(depth ?d)(time ?time2)(layer nil)(cell ?cell2))
        (test (eq ?l1 ?l2))
        (test (neq ?dep ?cell))
        (test (neq ?dep ?user))
        (test (neq ?cell ?user))
        (test (neq ?dep none))
=>
        (retract ?decide ?control)
        (assert (deciding_layer (depth ?dep)(cell ?cell)(user ?user)
                                (decision ?user)(location_id ?l1)
                                (timetag ?time1)))
        (assert (control (name current)(location_id ?l2)(previouse_id ?p2)
                        (track_id ?t1)(depth ?d)(time ?time2)(layer ?user)(cell ?cell2)))
        (assert (inferlayergoal name determine_layer status complete))
)
(defrule determine_layer_depthonly
        (subgoal name infer_layer status active)
        ?decide <- (deciding_layer (depth ?dep)(cell none)(user none)
                                (decision nil)(location_id ?l1)(timetag ?time1))
        ?controler <- (control (name current)(location_id ?l2)(previouse_id ?p2)
                                (track_id ?t2)(depth ?d)(time ?time2)(layer nil)
                                (cell ?cell2))
        ?depth <- (depth only)
```

```
        (test (eq ?l1 ?l2))
=>
        (retract ?decide ?controler ?depth)
        (assert (deciding_layer (depth ?dep)(cell none)(user none)
                                (decision ?dep)(location_id ?l1)(timetag ?time1)))
        (assert (control (name current)(location_id ?l2)(previouse_id ?p2)
                        (track_id ?t2)(depth ?d)(time ?time2)(layer ?dep)
                        (cell ?cell2)))
        (assert (inferlayergoal name determine_layer status complete))
)
(defrule determine_layer_dconly
        (subgoal name infer_layer status active)
        ?decide <- (deciding_layer (depth ?dep)(cell ?cell)(user none)
                                   (decision nil)(location_id ?l1)(timetag ?time1))
        ?controler <- (control (name current)(location_id ?l2)(previouse_id ?p2)
                                (track_id ?t2)(depth ?d)(time ?time2)(layer nil)
                                (cell ?cell2))
        (test (eq ?l1 ?l2))
        (test (eq ?dep ?cell))
=>
        (retract ?decide ?controler)
        (assert (deciding_layer (depth ?dep)(cell ?cell)(user none)(decision ?dep)
                                (location_id ?l1)(timetag ?time1)))
        (assert (control (name current)(location_id ?l2)(previouse_id ?p2)
                        (track_id ?t2)(depth ?d)(time ?time2)(layer ?dep)
```

```
                    (cell ?cell2)))
        (assert (inferlayergoal name determine_layer status complete))
)
(defrule determine_layer_dconlyno
        (subgoal name infer_layer status active)
        ?decide <- (deciding_layer (depth ?dep)(cell ?cell)(user none)
                                   (decision nil)(location_id ?l1)(timetag ?time1))
        ?controler <- (control (name current)(location_id ?l2)(previouse_id ?p2)
                                (track_id ?t2)(depth ?d)(time ?time2)(layer nil)
                                (cell ?cell2))
        (test (eq ?l1 ?l2))
        (test (neq ?dep ?cell))
=>
        (retract ?decide ?controler)
        (assert (deciding_layer (depth ?dep)(cell ?cell)(user none)
                                (decision ?cell)(location_id ?l1)(timetag ?time1)))
        (assert (control (name current)(location_id ?l2)(previouse_id ?p2)
                        (track_id ?t2)(depth ?d)(time ?time2)(layer none)
                        (cell ?cell2)))
        (assert (inferlayergoal name determine_layer status complete))
)
(defrule determine_layer_cunod
        (subgoal name infer_layer status active)
        ?decide <- (deciding_layer (depth none)(cell ?cell)(user ?user)
                                   (decision nil)(location_id ?l1)(timetag ?time1))
```

```
        ?controler <- (control (name current)(location_id ?l2)(previouse_id ?p2)
                                    (track_id ?t2)(depth ?d)(time ?time2)(layer nil)
                                    (cell ?cell2))
        (test (eq ?l1 ?l2))
=>
        (retract ?decide ?controler)
        (assert (deciding_layer (depth none)(cell ?cell)(user ?user)
                                    (decision ?user)(location_id ?l1)(timetag ?time1)))
        (assert (control (name current)(location_id ?l2)(previouse_id ?p2)
                        (track_id ?t2)(depth ?d)(time ?time2)(layer ?user)
                        (cell ?cell2)))
        (assert (inferlayergoal name determine_layer status complete))
)
(defrule determine_layer_cnod
        (subgoal name infer_layer status active)
        ?decide <- (deciding_layer (depth none)(cell ?cell)(user none)
                                    (decision nil)(location_id ?l1)
                                    (timetag ?time1))
        ?controler <- (control (name current)(location_id ?l2)(previouse_id ?p2)
                                    (track_id ?t2)(depth ?d)(time ?time2)(layer nil)
                                    (cell ?cell2))
        (test (eq ?l1 ?l2))
=>
        (retract ?decide ?controler)
        (assert (deciding_layer (depth none)(cell ?cell)(user none)
```

```
                    (decision none)(location_id ?l1)(timetag ?time1)))
        (assert (control (name current)(location_id ?l2)(previouse_id ?p2)
                    (track_id ?t2)(depth ?d)(time ?time2)(layer none)
                    (cell ?cell2)))
        (assert (inferlayergoal name determine_layer status complete))
)

;UPDATE PATHPLAN
;-----------------------------------------------------------------------------------------------------------
;It was discovered that it is important to keep track of the actual path that
;is found, because it could be different than the path plan sent to the KBS
;from the model.  These following rules attempt to update a new template
;called "new_pathplan" for each track.

;NO UPDATE OF PATHPLAN
;-----------------------------------------------------------------------------------------------------------
;For the occasion that there is no path update this rule will fire.
(defrule layertype_same
        (subgoal name infer_layer status active)
        ?controlnew <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                    (track_id ?t1)(depth ?d1)(time ?time1)(layer ?lay1)
                    (cell ?cell1)(state nil))
        (control (name done)(location_id ?l2)(previouse_id ?p2)(track_id ?t2)
                (depth ?d2)(time ?time2)(layer ?lay2)(cell ?cell2))
        (test (eq ?p1 ?l2))
```

```
        (test (> ?time1 ?time2))
        (test (eq ?t1 ?t2))
        (test (eq ?lay1 ?lay2))
        (not (depth limit wakeup))
=>
        (retract ?controlnew)
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer ?lay1)
                        (cell ?cell1)(state done)))
        (assert (inferlayergoal name done_pathplan status complete))
)
(defrule layertype_same_limit
        (subgoal name infer_layer status active)
        ?controlnew <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                                (track_id ?t1)(depth ?d1)(time ?time1)(layer ?lay1)
                                (cell ?cell1)(state nil))
        (control (name done)(location_id ?l2)(previouse_id ?p2)(track_id ?t2)
                (depth ?d2)(time ?time2)(layer ?lay2)(cell ?cell2))
        (test (eq ?p1 ?l2))
        (test (> ?time1 ?time2))
        (test (eq ?t1 ?t2))
        (test (eq ?lay1 ?lay2))
        ?limit <- (depth limit wakeup)
=>
        (retract ?controlnew ?limit)
```

```
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer ?lay1)
                        (cell ?cell1)(state done)))
        (assert (inferlayergoal name done_pathplan status complete))
)

;UPDATE PATHPLAN
;-----------------------------------------------------------------------------------------------------------------
;For when the layer type has changed and we want to update the new pathplan.
;FIRST UPDATE a special case exist for the first update so it has its own rule.
(defrule update_path_first
        (subgoal name infer_layer status active)
        ?first <- (first time)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d1)(time ?time1)(layer ?lay1)(cell ?cell1))
=>
        (retract ?first)
        (assert (inferlayergoal name done_pathplan status active))
)

;LAYERTYPE CHANGE fired every time there is a change in layer from the previouse.
(defrule layertype_change
        (subgoal name infer_layer status active)
        (control (name current)(location_id ?l1)(previouse_id ?p1)(track_id ?t1)
                (depth ?d1)(time ?time1)(layer ?lay1)(cell ?cell1))
```

```
        (control (name done)(location_id ?l2)(previouse_id ?p2)(track_id ?t2)
                (depth ?d2)(time ?time2)(layer ?lay2)(cell ?cell2))
        (test (eq ?p1 ?l2))
        (test (> ?time1 ?time2))
        (test (eq ?t1 ?t2))
        (test (neq ?lay1 ?lay2))
=>
        (assert (inferlayergoal name done_pathplan status active))
)

;if the new layer is none
(defrule update_path_none
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer none)
                        (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(Le ?le)(GPe ?gpe)(Li ?li)(GPi ?gpi)
                         (La ?la)(OT ?ot)(IC ?ic)(track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal ?control)
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer none)
                        (cell ?cell1)(state none)))
        (assert (inferlayergoal name done_pathplan status complete)))
```

```
;if the new layer is ST
(defrule update_path_ST
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                             (track_id ?t1)(depth ?d1)(time ?time1)(layer ST)
                             (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST 1000)(STstate nil)(Le 1000)(Lestate ?les)
                               (GPe 1000)(GPestate ?gpes)(Li 1000)(Listate ?lis)
                               (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                               (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                               (track_id ?t2))
        (test (eq ?t1 ?t2))
        (not (depth limit wakeup))
=>
        (retract ?goal)
        (retract ?control)
        (retract ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                         (track_id ?t1)(depth ?d1)(time ?time1)(layer ST)
                         (cell ?cell1)(state solid)))
        (assert (new_pathplan (ST ?d1)(STstate solid)(Le 1000)(Lestate ?les)
                              (GPe 1000)(GPestate ?gpes)(Li 1000)(Listate ?lis)
                              (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                              (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
```

```
                                (track_id ?t2)))
)

;if allready assigned ST
(defrule update_path_ST_skip
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                                (track_id ?t1)(depth ?d1)(time ?time1)(layer ST)
                                (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(Le ?le)(GPe ?gpe)(Li ?li)(GPi ?gpi)
                                        (La ?la)(OT ?ot)(IC ?ic)(track_id ?t2))
        (test (> ?d1 ?st))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal ?control)
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                                (track_id ?t1)(depth ?d1)(time ?time1)(layer ST)
                                (cell ?cell1)(state done)))
        (assert (inferlayergoal name done_pathplan status complete))
)

;if the new layer is ST and limit depth wakeup
(defrule update_path_ST_limit
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?lim <- (depth limit wakeup)
```

```
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                             (track_id ?t1)(depth ?d1)(time ?time1)(layer ST)
                             (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST 1000)(STstate nil)(Le 1000)(Lestate ?les)
                               (GPe 1000)(GPestate ?gpes)(Li 1000)(Listate ?lis)
                               (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                               (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                               (track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal ?lim ?control ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                         (track_id ?t1)(depth ?d1)(time ?time1)(layer ST)
                         (cell ?cell1)(state fuzz)))
        (assert (new_pathplan (ST ?d1)(STstate fuzz)(Le 1000)(Lestate ?les)
                              (GPe 1000)(GPestate ?gpes)(Li 1000)(Listate ?lis)
                              (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                              (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                              (track_id ?t2)))
)

;if the new layer is Le
(defrule update_path_Le
        ?goal <- (inferlayergoal name done_pathplan status active)
```

```
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                             (track_id ?t1)(depth ?d1)(time ?time1)(layer Le)
                             (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le 1000)(Lestate nil)
                               (GPe 1000)(GPestate ?gpes)(Li 1000)(Listate ?lis)
                               (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                                  (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                                  (track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal)
        (retract ?control)
        (retract ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                         (track_id ?t1)(depth ?d1)(time ?time1)(layer Le)
                         (cell ?cell1)(state solid)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?d1)(Lestate solid)
                              (GPe 1000)(GPestate ?gpes)(Li 1000)(Listate ?lis)
                              (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                              (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                              (track_id ?t2)))
)
```

```
;if allready assigned Le
(defrule update_path_Le_skip
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer Le)
                        (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(Le ?le)(GPe ?gpe)(Li ?li)(GPi ?gpi)
                            (La ?la)(OT ?ot)(IC ?ic)(track_id ?t2))
        (test (> ?d1 ?le))
=>
        (retract ?goal ?control)
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer Le)
                        (cell ?cell1)(state done)))
        (assert (inferlayergoal name done_pathplan status complete))
)

;if the new layer is Le and limit depth wakeup
(defrule update_path_Le_limit
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?lim <- (depth limit wakeup)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer Le)
                        (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le 1000)(Lestate nil)
```

```
                                (GPe 1000)(GPestate ?gpes)(Li 1000)(Listate ?lis)
                                (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                                (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                                (track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal ?lim ?control ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer Le)
                        (cell ?cell1)(state fuzz)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?d1)(Lestate fuzz)
                            (GPe 1000)(GPestate ?gpes)(Li 1000)(Listate ?lis)
                            (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                            (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                            (track_id ?t2)))
)

;if the new layer is GPe
(defrule update_path_GPe
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                            (track_id ?t1)(depth ?d1)(time ?time1)(layer GPe)
                            (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
```

```
                                (GPe 1000)(GPestate ?gpes)(Li 1000)(Listate ?lis)
                                (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                                (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                                (track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal)
        (retract ?control)
        (retract ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer GPe)
                        (cell ?cell1)(state solid)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                                (GPe ?d1)(GPestate solid)(Li 1000)(Listate ?lis)
                                (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                                (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                                (track_id ?t2)))
)

;if allready assigned GPe
(defrule update_path_GPe_skip
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                                (track_id ?t1)(depth ?d1)(time ?time1)(layer GPe)
```

```
                    (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(Le ?le)(GPe ?gpe)(Li ?li)(GPi ?gpi)
                            (La ?la)(OT ?ot)(IC ?ic)(track_id ?t2))
        (test (> ?d1 ?gpe))
=>
        (retract ?goal ?control)
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                    (track_id ?t1)(depth ?d1)(time ?time1)(layer GPe)
                    (cell ?cell1)(state done)))
        (assert (inferlayergoal name done_pathplan status complete))
)

;if the new layer is GPe and limit depth wakeup
(defrule update_path_GPe_limit
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?lim <- (depth limit wakeup)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                    (track_id ?t1)(depth ?d1)(time ?time1)(layer GPe)
                    (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                            (GPe 1000)(GPestate nil)(Li 1000)(Listate ?lis)
                            (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                            (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                            (track_id ?t2))
        (test (eq ?t1 ?t2))
```

```
=>
        (retract ?goal ?lim ?control ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer GPe)
                        (cell ?cell1)(state fuzz)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                            (GPe ?d1)(GPestate fuzz)(Li 1000)(Listate ?lis)
                            (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                            (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                            (track_id ?t2)))
)

;if the new layer is Li
(defrule update_path_Li
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                            (track_id ?t1)(depth ?d1)(time ?time1)(layer Li)
                            (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                                (GPe ?gpe)(GPestate ?gpes)(Li 1000)(Listate nil)
                                (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                                (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                                (track_id ?t2))
        (test (eq ?t1 ?t2))
```

```
=>
        (retract ?goal)
        (retract ?control)
        (retract ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer Li)
                        (cell ?cell1)(state solid)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                                (GPe ?gpe)(GPestate ?gpes)(Li ?d1)(Listate solid)
                                (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                                (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                                (track_id ?t2)))
)

;if allready assigned Li
(defrule update_path_Li_skip
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer Li)
                        (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(Le ?le)(GPe ?gpe)(Li ?li)(GPi ?gpi)
                                (La ?la)(OT ?ot)(IC ?ic)(track_id ?t2))
        (test (> ?d1 ?li))
=>
```

```
        (retract ?goal ?control)
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer Li)
                        (cell ?cell1)(state done)))
        (assert (inferlayergoal name done_pathplan status complete))
)

;if the new layer is Li and limit depth wakeup
(defrule update_path_Li_limit
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?lim <- (depth limit wakeup)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer Li)
                        (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                        (GPe ?gpe)(GPestate ?gpes)(Li 1000)(Listate nil)
                        (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                        (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                        (track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal ?lim ?control ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer Li)
```

```
                    (cell ?cell1)(state fuzz)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                    (GPe ?gpe)(GPestate ?gpes)(Li ?d1)(Listate fuzz)
                    (GPi 1000)(GPistate ?gpis)(La 1000)(Lastate ?las)
                    (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                    (track_id ?t2)))
)

;if the new layer is GPi
(defrule update_path_GPi
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                    (track_id ?t1)(depth ?d1)(time ?time1)(layer GPi)
                    (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                    (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                    (GPi 1000)(GPistate nil)(La 1000)(Lastate ?las)
                    (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                    (track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal)
        (retract ?control)
        (retract ?path)
        (assert (inferlayergoal name done_pathplan status complete))
```

```
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer GPi)
                        (cell ?cell1)(state solid)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                                (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                                (GPi ?d1)(GPistate solid)(La 1000)(Lastate ?las)
                                (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                                (track_id ?t2)))
)

;if allready assigned GPi
(defrule update_path_GPi_skip
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                            (track_id ?t1)(depth ?d1)(time ?time1)(layer GPi)
                            (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(Le ?le)(GPe ?gpe)(Li ?li)(GPi ?gpi)
                                (La ?la)(OT ?ot)(IC ?ic)(track_id ?t2))
        (test (> ?d1 ?gpi))
=>
        (retract ?goal ?control)
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer GPi)
                        (cell ?cell1)(state done)))
        (assert (inferlayergoal name done_pathplan status complete))
```

```
)

;if the new layer is GPi and limit depth wakeup
(defrule update_path_GPi_limit
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?lim <- (depth limit wakeup)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer GPi)
                        (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                            (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                            (GPi 1000)(GPistate nil)(La 1000)(Lastate ?las)
                            (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                            (track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal ?lim ?control ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer GPi)
                        (cell ?cell1)(state fuzz)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                            (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                            (GPi ?d1)(GPistate fuzz)(La 1000)(Lastate ?las)
                            (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
```

```
                                (track_id ?t2)))
)

;if the new layer is La
(defrule update_path_La
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                                (track_id ?t1)(depth ?d1)(time ?time1)(layer La)
                                (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                                        (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                                        (GPi ?gpi)(GPistate ?gpis)(La 1000)(Lastate nil)
                                        (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                                        (track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal)
        (retract ?control)
        (retract ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                                (track_id ?t1)(depth ?d1)(time ?time1)(layer La)
                                (cell ?cell1)(state solid)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                                        (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
```

```
                                (GPi ?gpi)(GPistate ?gpis)(La ?d1)(Lastate solid)
                                (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                                (track_id ?t2)))
)

;if allready assigned La
(defrule update_path_La_skip
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer La)
                        (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(Le ?le)(GPe ?gpe)(Li ?li)(GPi ?gpi)
                        (La ?la)(OT ?ot)(IC ?ic)(track_id ?t2))
        (test (> ?d1 ?la))
=>
        (retract ?goal ?control)
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer La)
                        (cell ?cell1)(state done)))
        (assert (inferlayergoal name done_pathplan status complete))
)

;if the new layer is Le and limit depth wakeup
(defrule update_path_La_limit
        ?goal <- (inferlayergoal name done_pathplan status active)
```

```
        ?lim <- (depth limit wakeup)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                             (track_id ?t1)(depth ?d1)(time ?time1)(layer La)
                             (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                               (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                               (GPi ?gpi)(GPistate ?gpis)(La 1000)(Lastate nil)
                               (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                               (track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal ?lim ?control ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                         (track_id ?t1)(depth ?d1)(time ?time1)(layer La)
                         (cell ?cell1)(state fuzz)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                              (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                              (GPi ?gpi)(GPistate ?gpis)(La ?d1)(Lastate fuzz)
                              (OT 1000)(OTstate ?ots)(IC 1000)(ICstate ?ics)
                              (track_id ?t2)))
)

;if the new layer is OT
(defrule update_path_OT
```

```
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                             (track_id ?t1)(depth ?d1)(time ?time1)(layer OT)
                             (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                               (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                               (GPi ?gpi)(GPistate ?gpis)(La ?la)(Lastate ?las)
                               (OT 1000)(OTstate nil)(IC 1000)(ICstate ?ics)
                               (track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal)
        (retract ?control)
        (retract ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                         (track_id ?t1)(depth ?d1)(time ?time1)(layer OT)
                         (cell ?cell1)(state solid)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                               (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                               (GPi ?gpi)(GPistate ?gpis)(La ?la)(Lastate ?las)
                               (OT ?d1)(OTstate solid)(IC 1000)(ICstate ?ics)
                               (track_id ?t2)))
)
```

```
;if allready assigned OT
(defrule update_path_OT_skip
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer OT)
                        (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(Le ?le)(GPe ?gpe)(Li ?li)(GPi ?gpi)
                            (La ?la)(OT ?ot)(IC ?ic)(track_id ?t2))
        (test (> ?d1 ?ot))
=>
        (retract ?goal ?control)
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer OT)
                        (cell ?cell1)(state done)))
        (assert (inferlayergoal name done_pathplan status complete))
)

;if the new layer is OT and limit depth wakeup
(defrule update_path_OT_limit
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?lim <- (depth limit wakeup)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer OT)
                        (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
```

```
                                (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                                (GPi ?gpi)(GPistate ?gpis)(La ?la)(Lastate ?las)
                                (OT 1000)(OTstate nil)(IC 1000)(ICstate ?ics)
                                (track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal ?lim ?control ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer OT)
                        (cell ?cell1)(state fuzz)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                            (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                            (GPi ?gpi)(GPistate ?gpis)(La ?la)(Lastate ?las)
                            (OT ?d1)(OTstate fuzz)(IC 1000)(ICstate ?ics)
                            (track_id ?t2)))
)

;if the new layer is IC
(defrule update_path_IC
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                            (track_id ?t1)(depth ?d1)(time ?time1)(layer IC)
                            (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
```

```
                                (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                                (GPi ?gpi)(GPistate ?gpis)(La ?la)(Lastate ?las)
                                (OT ?ot)(OTstate ?ots)(IC 1000)(ICstate nil)(track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?goal)
        (retract ?control)
        (retract ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer IC)
                        (cell ?cell1)(state solid)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                                (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                                (GPi ?gpi)(GPistate ?gpis)(La ?la)(Lastate ?las)
                                (OT ?ot)(OTstate ?ots)(IC ?d1)(ICstate solid)
                                (track_id ?t2)))
)

;if allready assigned IC
(defrule update_path_IC_skip
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer IC)
                        (cell ?cell1)(state done))
```

```
        ?path <- (new_pathplan (ST ?st)(Le ?le)(GPe ?gpe)(Li ?li)(GPi ?gpi)
                                (La ?la)(OT ?ot)(IC ?ic)(track_id ?t2))
        (test (> ?d1 ?ic))
=>
        (retract ?goal ?control)
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                        (track_id ?t1)(depth ?d1)(time ?time1)(layer IC)
                        (cell ?cell1)))
        (assert (inferlayergoal name done_pathplan status complete))
)

;if the new layer is IC and limit depth wakeup
(defrule update_path_IC_limit
        ?goal <- (inferlayergoal name done_pathplan status active)
        ?lim <- (depth limit wakeup)
        ?control <- (control (name current)(location_id ?l1)(previouse_id ?p1)
                            (track_id ?t1)(depth ?d1)(time ?time1)(layer ST)
                            (cell ?cell1)(state nil))
        ?path <- (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                                (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                                (GPi ?gpi)(GPistate ?gpis)(La ?la)(Lastate ?las)
                                (OT ?ot)(OTstate ?ots)(IC 1000)(ICstate nil)
                                (track_id ?t2))
        (test (eq ?t1 ?t2))
=>
```

```
        (retract ?goal ?lim ?control ?path)
        (assert (inferlayergoal name done_pathplan status complete))
        (assert (control (name done)(location_id ?l1)(previouse_id ?p1)
                         (track_id ?t1)(depth ?d1)(time ?time1)(layer IC)
                         (cell ?cell1)(state fuzz)))
        (assert (new_pathplan (ST ?st)(STstate ?sts)(Le ?le)(Lestate ?les)
                              (GPe ?gpe)(GPestate ?gpes)(Li ?li)(Listate ?lis)
                              (GPi ?gpi)(GPistate ?gpis)(La ?la)(Lastate ?las)
                              (OT ?ot)(OTstate ?ots)(IC ?d1)(ICstate fuzz)
                              (track_id ?t2)))
)
```

D.16. Track Planning

```
;       ---------------------------------------------------------------------------------------------------
;                       Track Planning
;       Program:        Pallidotomy Version 6.0
;       Author:         Linda Harley and Dr. N Baker
;       Date:           September 4, 2003
;       File:           trackplanning.clp
;       Updates:        Sep 4           Created defrule "cleanup"
;       ---------------------------------------------------------------------------------------------------
;CLEANUP
;---------------------------------------------------------------------------------------------------------
;In order to have multiple tracks be executed some process rules needs
;to be removed in order to go through the cycle of executions again.
(defrule cleanup_track
        (goal name trackplanning status active)
        ?onetrack <- (goal name onetrack status complete)
        ?decide <- (deciding_layer (depth nil)(cell nil)(user nil)(decision nil)
                                (location_id nil)(timetag nil))
        ?initialize <- (goal name initialize status complete)
        ?expect <- (expecting_layer (name nil)(location_id nil)
                                 (previouse_id nil)(timetag 0))
        ?cont <- (control (name nil)(location_id nil)(previouse_id nil)
                        (track_id ?t)(depth nil)(time 0)(layer nil)(cell nil))
=>
        (retract ?onetrack)
```

```
        (retract ?decide)
        (retract ?initialize)
        (retract ?expect)
        (retract ?cont)
        (assert (trackgoal name cleanup_track status complete))
)
(defrule tp_calculate_layer_thickness
        (goal name trackplanning status active)
        (new_pathplan (ST ?st)(Le ?le)(GPe ?gpe)(Li ?li)(GPi ?gpi)(La ?la)
                (OT ?ot)(IC ?ic)(track_id ?t1))
        ?layer <- (layer_thick (ST 1000)(GPe 1000)(GPi 1000)(track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (assert (layer_thick
                (ST (- ?le ?st))
                (GPe (- ?li ?gpe))
                (GPi (- ?la ?gpi))
                (track_id ?t1)))
        (assert (tpgoal name layerthick status complete))
)
(defrule tp_ST_anterior_lateral
        (goal name trackplanning status active)
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st)(GPe ?gpe1)(GPi ?gpi1)(track_id ?t1))
        ?apml <- (apml_assign (ST nil)(GPe ?gpe2)(GPi ?gpi2)(OT ?ot2)(IC ?ic2)
```

```
                                (status nil)(track_id ?t2))
        (test (eq ?t1 ?t2))
        (test (> ?st 8.0))
        (test (< ?st 100))
        (test (neq ?st 0))
=>
        (retract ?apml)
        (assert (apml_assign (ST anterior)(GPe ?gpe2)(GPi ?gpi2)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2)))
        (assert (apml_assign (ST lateral)(GPe ?gpe2)(GPi ?gpi2)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2)))
)
(defrule tp_ST_posterior_medial
        (goal name trackplanning status active)
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(track_id ?t1))
        ?apml <- (apml_assign (ST nil)(GPe ?gpe2)(GPi ?gpi2)(OT ?ot2)
                                        (IC ?ic2)(status nil)(track_id ?t2))
        (test (eq ?t1 ?t2))
        (test (<= ?st1 8))
        (test (neq ?st1 0))
=>
        (retract ?apml)
        (assert (apml_assign (ST posterior)(GPe ?gpe2)(GPi ?gpi2)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2)))
```

```
        (assert (apml_assign (ST medial)(GPe ?gpe2)(GPi ?gpi2)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2)))
)

(defrule tp_ST_none
        (goal name trackplanning status active)
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(track_id ?t1))
        ?apml <- (apml_assign (ST nil)(GPe ?gpe2)(GPi ?gpi2)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2))
        (new_pathplan (ST 1000)(Le ?le3)(GPe ?gpe3)(Li ?li3)(GPi ?gpi3)
                                (La ?la3)(OT ?ot3)(IC ?ic3)(track_id ?t3))
        (test (eq ?t1 ?t2 ?t3))
=>
        (retract ?apml)
        (assert (apml_assign (ST none)(GPe ?gpe2)(GPi ?gpi2)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2)))
)
(defrule tp_GPe_anterior
        (goal name trackplanning status active)
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(track_id ?t1))
        ?apml <- (apml_assign (ST ?st2)(GPe nil)(GPi ?gpi2)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2))
        (test (> ?gpe1 6))
```

```
        (test (< ?gpe1 100))
        (test (eq ?t1 ?t2))
        (test (neq ?gpe1 0))
=>
        (retract ?apml)
        (assert (apml_assign (ST ?st2)(GPe anterior)(GPi ?gpi2)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2)))
)
(defrule tp_GPe_central
        (goal name trackplanning status active)
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(track_id ?t1))
        ?apml <- (apml_assign (ST ?st2)(GPe nil)(GPi ?gpi2)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2))
        (test (and (<= ?gpe1 6)
                (>= ?gpe1 4)))
        (test (eq ?t1 ?t2))
        (test (neq ?gpe1 0))
=>
        (retract ?apml)
        (assert (apml_assign (ST ?st2)(GPe central)(GPi ?gpi2)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2)))
)
(defrule tp_GPe_none
        (goal name trackplanning status active)
```

```
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(track_id ?t1))
        ?apml <- (apml_assign (ST ?st2)(GPe nil)(GPi ?gpi2)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2))
        (new_pathplan (ST ?st3)(Le ?le3)(GPe ?gpe3)(Li ?li3)(GPi ?gpi3)
                (La ?la3)(OT ?ot3)(IC ?ic3)(track_id ?t3))
        (test (eq ?t1 ?t2 ?t3))
        (test (or (= ?gpe3 1000)
                (< ?gpe1 4)))
=>
        (retract ?apml)
        (assert (apml_assign (ST ?st2)(GPe none)(GPi ?gpi2)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2)))
)
(defrule tp_GPi_apl
        (goal name trackplanning status active)
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(track_id ?t1))
        ?apml <- (apml_assign (ST ?st2)(GPe ?gpe2)(GPi nil)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2))
        (test (eq ?t1 ?t2))
        (test (<= ?gpi1 4))
        (test (neq ?gpi1 0))
=>
        (retract ?apml)
```

```
        (assert (apml_assign (ST ?st2)(GPe ?gpe2)(GPi anterior)(OT ?ot2)

                                (IC ?ic2)(status nil)(track_id ?t2)))
        (assert (apml_assign (ST ?st2)(GPe ?gpe2)(GPi posterior)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2)))
        (assert (apml_assign (ST ?st2)(GPe ?gpe2)(GPi lateral)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2)))
)
(defrule tp_GPi_medial
        (goal name trackplanning status active)
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(track_id ?t1))
        ?apml <- (apml_assign (ST ?st2)(GPe ?gpe2)(GPi nil)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2))
        (test (eq ?t1 ?t2))
        (test (and (> ?gpi1 4)
                (< ?gpi1 6)))
        (test (neq ?gpi1 0))
=>
        (retract ?apml)
        (assert (apml_assign (ST ?st2)(GPe ?gpe2)(GPi medial)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2)))
)
(defrule tp_GPi_central
        (goal name trackplanning status active)
```

```
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(track_id ?t1))
        ?apml <- (apml_assign (ST ?st2)(GPe ?gpe2)(GPi nil)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2))
        (test (eq ?t1 ?t2))
        (test (and (>= ?gpi1 6)
                (< ?gpi1 100)))
=>
        (retract ?apml)
        (assert (apml_assign (ST ?st2)(GPe ?gpe2)(GPi central)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2)))
)
(defrule tp_GPi_none
        (goal name trackplanning status active)
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(track_id ?t1))
        ?apml <- (apml_assign (ST ?st2)(GPe ?gpe2)(GPi nil)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2))
        (new_pathplan (ST ?st3)(Le ?le3)(GPe ?gpe3)(Li ?li3)(GPi 1000)
                        (La ?la3)(OT ?ot3)(IC ?ic3)(track_id ?t3))
        (test (eq ?t1 ?t2 ?t3))
=>
        (retract ?apml)
        (assert (apml_assign (ST ?st2)(GPe ?gpe2)(GPi none)(OT ?ot2)
                                (IC ?ic2)(status nil)(track_id ?t2)))
```

```
)
(defrule tp_OT_positive
        (goal name trackplanning status active)
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(track_id ?t1))
        ?apml <- (apml_assign (ST ?st2)(GPe ?gpe2)(GPi ?gpi2)(OT nil)
                                (IC ?ic2)(status nil)(track_id ?t2))
        (new_pathplan (ST ?st3)(Le ?le3)(GPe ?gpe3)(Li ?li3)(GPi ?gpi3)
                        (La ?la3)(OT ?ot3)(IC ?ic3)(track_id ?t3))
        (test (eq ?t1 ?t2 ?t3))
        (test (neq ?ot3 1000))
=>
        (retract ?apml)
        (assert (apml_assign (ST ?st2)(GPe ?gpe2)(GPi ?gpi2)(OT positive)
                                (IC ?ic2)(status nil)(track_id ?t2)))
)
(defrule tp_OT_negative
        (goal name trackplanning status active)
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(track_id ?t1))
        ?apml <- (apml_assign (ST ?st2)(GPe ?gpe2)(GPi ?gpi2)(OT nil)
                                (IC ?ic2)(status nil)(track_id ?t2))
        (new_pathplan (ST ?st3)(Le ?le3)(GPe ?gpe3)(Li ?li3)(GPi ?gpi3)
                        (La ?la3)(OT 1000)(IC ?ic3)(track_id ?t3))
        (test (eq ?t1 ?t2 ?t3))
```

```
=>
        (retract ?apml)
        (assert (apml_assign (ST ?st2)(GPe ?gpe2)(GPi ?gpi2)(OT negative)
                                (IC ?ic2)(status nil)(track_id ?t2)))
)
(defrule tp_IC_positive
        (goal name trackplanning status active)
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(track_id ?t1))
        ?apml <- (apml_assign (ST ?st2)(GPe ?gpe2)(GPi ?gpi2)(OT ?ot2)
                                (IC nil)(status nil)(track_id ?t2))
        (new_pathplan (ST ?st3)(Le ?le3)(GPe ?gpe3)(Li ?li3)(GPi ?gpi3)
                        (OT ?ot3)(IC ?ic3)(track_id ?t3))
        (test (eq ?t1 ?t2 ?t3))
        (test (neq ?ic3 1000))
=>
        (retract ?apml)
        (assert (apml_assign (ST ?st2)(GPe ?gpe2)(GPi ?gpi2)(OT ?ot2)
                                (IC positive)(status nil)(track_id ?t2)))
)
(defrule tp_IC_negative
        (goal name trackplanning status active)
        (tpgoal name layerthick status complete)
        (layer_thick (ST ?st)(GPe ?gpe)(GPi ?gpi)(track_id ?t))
        ?apml <- (apml_assign (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(OT ?ot1)
```

```
                                (IC nil)(status nil)(track_id ?t1))
        (new_pathplan (ST ?st2)(Le ?le2)(GPe ?gpe2)(Li ?li2)(GPi ?gpi2)
                        (OT ?ot2)(IC 1000)(track_id ?t2))
        (test (eq ?t ?t1 ?t2))
=>
        (retract ?apml)
        (assert (apml_assign (ST ?st1)(GPe ?gpe1)(GPi ?gpi1)(OT ?ot1)
                                (IC negative)(status nil)(track_id ?t1)))
)


(defrule tp_done_apml_assign
        (goal name trackplanning status active)
        (not (apml_assign (ST nil)(GPe nil)(GPi nil)(OT nil)(IC nil)
                        (status nil)(track_id ?t1)))
=>
        (assert (tpgoal name apml_assign status complete))
        (assert (tpgoal name plan_track status active))
)
(defrule tp_propose_no_GPi
        (tpgoal name plan_track status active)
        ?apml <- (apml_assign (ST ?st)(GPe ?gpe)(GPi none)(OT ?ot)(IC ?ic)
                        (status nil)(track_id ?t))
        (test (or (neq ?st none)
                (neq ?gpe none)))
```

```
=>
        (retract ?apml)
        (assert (output (type M)(format message)(variable nil)
                        (text-field "No GPi found.  Check equipment.
                        Check for hemorrhaging and start new track.")
                        (depth nil)(timetag nil)))
        (assert (goal interface_initiated status active))
        (assert (apml_assign (ST ?st)(GPe ?gpe)(GPi none)(OT ?ot)(IC ?ic)
                             (status done)(track_id ?t)))
)


(defrule tp_propose_in
        ?goal <- (tpgoal name plan_track status active)
        ?answer <- (answer_received (format message))
=>
        (retract ?goal ?answer)
        (assert (tpgoal name plan_track status complete))
)
(defrule tp_propose_IC_pos
        (tpgoal name plan_track status active)
        ?apml <- (apml_assign (ST ?st)(GPe ?gpe)(GPi ?gpi)(OT ?ot)(IC positive)
                              (status nil)(track_id ?t))
=>
        (retract ?apml)
```

```
        (assert (output (type M)(format message)(variable nil)
                        (text-field "IC found.  AP defined.  Move 2mm
                        medial or lateral")(depth nil)(timetag nil)))
        (assert (goal interface_initiated status active))
        (assert (apml_assign (ST ?st)(GPe ?gpe)(GPi ?gpi)(OT ?ot)(IC positive)
                             (status done)(track_id ?t)))
)
(defrule tp_propose_OTpos_ICpos
        (tpgoal name plan_track status active)
        ?apml <- (apml_assign (ST ?st)(GPe ?gpe)(GPi ?gpi)(OT positive)
                              (IC positive)(status nil)(track_id ?t))
=>
        (retract ?apml)
        (assert (output (type M)(format message)(variable nil)
                        (text-field "Both OT and IC positive.  Move
                        track medial, lateral or posterior.")
                        (depth nil)(timetag nil)))
        (assert (goal interface_initiated status active))
        (assert (apml_assign (ST ?st)(GPe ?gpe)(GPi ?gpi)(OT positive)
                             (IC positive)(status done)(track_id ?t)))
)
(defrule tp_propose_OTpos_ICneg
        (tpgoal name plan_track status active)
        ?apml <- (apml_assign (ST ?st)(GPe ?gpe)(GPi ?gpi)(OT positive)
                              (IC negative)(status nil)(track_id ?t))
```

```
=>
        (retract ?apml)
        (assert (output (type M)(format message)(variable nil)
                        (text-field "Move 2mm posterior.")))
        (assert (goal interface_initiated status active))
        (assert (apml_assign (ST ?st)(GPe ?gpe)(GPi ?gpi)(OT positive)
                             (IC negative)(status done)(track_id ?t)))
)
(defrule tp_propose_no_ST
        (tpgoal name plan_track status active)
        ?apml <- (apml_assign (ST none)(GPe ?gpe)(GPi ?gpi)(OT negative)
                              (IC negative)(status nil)(track_id ?t))
=>
        (retract ?apml)
        (assert (output (type M)(format message)(variable nil)
                        (text-field "No Striatum found.
                        Check equipment.")))
        (assert (goal interface_initiated status active))
        (assert (apml_assign (ST none)(GPe ?gpe)(GPi ?gpi)(OT negative)
                             (IC negative)(status done)(track_id ?t)))
)
(defrule tp_propose_no_GPe
        (tpgoal name plan_track status active)
        ?apml <- (apml_assign (ST ?st)(GPe none)(GPi ?gpi)(OT negative)
                              (IC negative)(status nil)(track_id ?t))
```

```
        (test (or (neq ?st none)
                  (neq ?gpi none)))
=>
        (retract ?apml)
        (assert (output (type M)(format message)(variable nil)
                        (text-field "No GPe found.  Check equipment.")))
        (assert (goal interface_initiated status active))
        (assert (apml_assign (ST ?st)(GPe none)(GPi ?gpi)(OT negative)
                             (IC negative)(status done)(track_id ?t)))
)
(defrule tp_propose_central
        (tpgoal name plan_track status active)
        ?apml <- (apml_assign (ST ?st)(GPe central)(GPi central)(OT negative)
                              (IC negative)(status nil)(track_id ?t))
=>
        (retract ?apml)
        (assert (output (type M)(format message)(variable nil)
                        (text-field "Move 2mm posterior.")))
        (assert (goal interface_initiated status active))
        (assert (apml_assign (ST ?st)(GPe central)(GPi central)(OT negative)
                             (IC negative)(status done)(track_id ?t)))
)
(defrule tp_propose_Anterior_stmatch
        (tpgoal name plan_track status active)
        ?apml <- (apml_assign (ST ?st)(GPe ?gpe)(GPi ?gpi)(OT negative)
```

```
                    (IC negative)(status nil)(track_id ?t))
        (test (or (eq ?st ?gpe ?gpi)
                  (eq ?st ?gpe)
                  (eq ?st ?gpi)))
        (test (eq ?st anterior))
=>
        (retract ?apml)
        (assert (output (type M)(format message)(variable nil)
                        (text-field "Move 3-4mm posterior.")))
        (assert (goal interface_initiated status active))
        (assert (apml_assign (ST ?st)(GPe ?gpe)(GPi ?gpi)(OT negative)
                             (IC negative)(status done)(track_id ?t)))
)
(defrule tp_propose_Anterior_nostmatch
        (tpgoal name plan_track status active)
        ?apml <- (apml_assign (ST ?st)(GPe anterior)(GPi anterior)
                              (OT negative)(IC negative)
                              (status nil)(track_id ?t))
        (test (neq ?st anterior))
=>
        (retract ?apml)
        (assert (apml_assign (ST ?st)(GPe anterior)(GPi anterior)
                             (OT negative)(IC negative)
                             (status done)(track_id ?t)))
        (assert (output (type M)(format message)(variable nil)
```

```
                    (text-field "Move 3-4mm posterior.")))
        (assert (goal interface_initiated status active)))
(defrule tp_propose_lateral
        (tpgoal name plan_track status active)
        ?apml <- (apml_assign (ST lateral)(GPe ?gpe)(GPi lateral)
                              (OT negative)(IC negative)
                              (status nil)(track_id ?t))
=>
        (retract ?apml)
        (assert (apml_assign (ST lateral)(GPe ?gpe)(GPi lateral)
                             (OT negative)(IC negative)
                             (status done)(track_id ?t)))
        (assert (output (type M)(format message)(variable nil)
                        (text-field "Probe is lateral.  Move anterior,
                        medial or posterior.")))
        (assert (goal interface_initiated status active)))
(defrule tp_propose_Posterior
        (tpgoal name plan_track status active)
        ?apml <- (apml_assign (ST posterior)(GPe ?gpe)(GPi posterior)
                              (OT negative)(IC negative)
                              (status nil)(track_id ?t))
=>
        (retract ?apml)
        (assert (apml_assign (ST posterior)(GPe ?gpe)(GPi posterior)
                             (OT negative)(IC negative)
```

```
                    (status done)(track_id ?t)))
        (assert (output (type M)(format message)(variable nil)
                    (text-field "Probe is posterior.
                    Move 1mm posterior to find Internal Capsule.")))
        (assert (goal interface_initiated status active)))
(defrule tp_propose_Medial
        (tpgoal name plan_track status active)
        ?apml <- (apml_assign (ST medial)(GPe ?gpe)(GPi medial)
                    (OT negative)(IC negative)
                    (status nil)(track_id ?t))
=>
        (retract ?apml)
        (assert (apml_assign (ST medial)(GPe ?gpe)(GPi medial)
                    (OT negative)(IC negative)
                    (status done)(track_id ?t)))
        (assert (output (type M)(format message)(variable nil)
                    (text-field "Probe is medial.  Either move
                    2mm lateral or complete mapping.")))
        (assert (goal interface_initiated status active)))
(defrule tp_propose_Nothing
        (tpgoal name plan_track status active)
        ?apml <- (apml_assign (ST ?st)(GPe ?gpe)(GPi ?gpi)(OT negative)
                    (IC negative)(status nil)(track_id ?t))
        (test (neq ?st ?gpe ?gpi))
=>
```

```
        (retract ?apml)
        (assert (apml_assign (ST ?st)(GPe ?gpe)(GPi ?gpi)(OT negative)
                             (IC negative)(status done)(track_id ?t)))
        (assert (output (type M)(format message)(variable nil)
                        (text-field "KBS unable to suggest where
                        to place the next track.")))
        (assert (goal interface_initiated status active))
)
(defrule tp_cleanup
        (tpgoal name plan_track status complete)
        (apml_assign (ST ?st)(GPe ?gpe)(GPi ?gpi)(OT ?ot)
                     (IC ?ic)(status done)(track_id ?t1))
        ?apml <- (apml_assign (ST ?st2)(GPe ?gpe2)(GPi ?gpi2)(OT ?ot2)
                              (IC ?ic2)(status nil)(track_id ?t2))
        (test (eq ?t1 ?t2))
=>
        (retract ?apml)
)
(defrule tp_done_planning
        (tpgoal name plan_track status complete)
        (not (apml_assign (ST ?st)(GPe ?gpe)(GPi ?gpi)(OT ?ot)
                          (IC ?ic)(status nil)(track_id ?t1)))
=>
        (assert (tpgoal name cleanup status complete))
)
```

APPENDIX E

USER'S MANUAL FOR CLIPS

E.1. Introduction

This user's manual refers to the KBS program for the Onetrack system. For the purpose of this discussion the KBS will run in the CLIPS environment. The CLIPS software can be downloaded from this website http://www.ghg.net/clips/CLIPS.html. Follow the instructions on the website to install this software on the user's computer. All the necessary files for the KBS are shown in Appendix D. It is best to use the Notepad software to create these files. The appropriate name for each file is indicated at the start of each new file in Appendix D. If no errors occur during loading then the program should run flawlessly. These files are also available on a CD that is in the back folder of this book. Copy the files to the desired directory on the user's computer.

E.2. Setup

Two things need to be checked before the KBS can be executed in the CLIPS environment. First check the "pal.bat" file, by opening the file in Notepad. Ensure that the directory links to all of the .clp files are correct, by looking at the file names and ensuring that they all end with ".clp". In the event that the .bat file is in the same directory as the .clp files there should be no need to change the directories. Secondly open the "template.clp" file and under the heading of GLOBALS ensure that the following is true:

```
(defglobal ?*output_device* = t)
;(defglobal ?*output_device* = wdialog)
```

Note that the semi colon is in front of the second line of text. This implies that this line of text is inactive. If the KBS is to be executed in the Onetrack system environment then the semi colon should be placed in front of the first line of text. Now that these two files are correct, the KBS can be loaded into the CLIPS program.

E.3. Loading the KBS

Upon opening the CLIPS program the screen indicated in Figure E.1 appears.

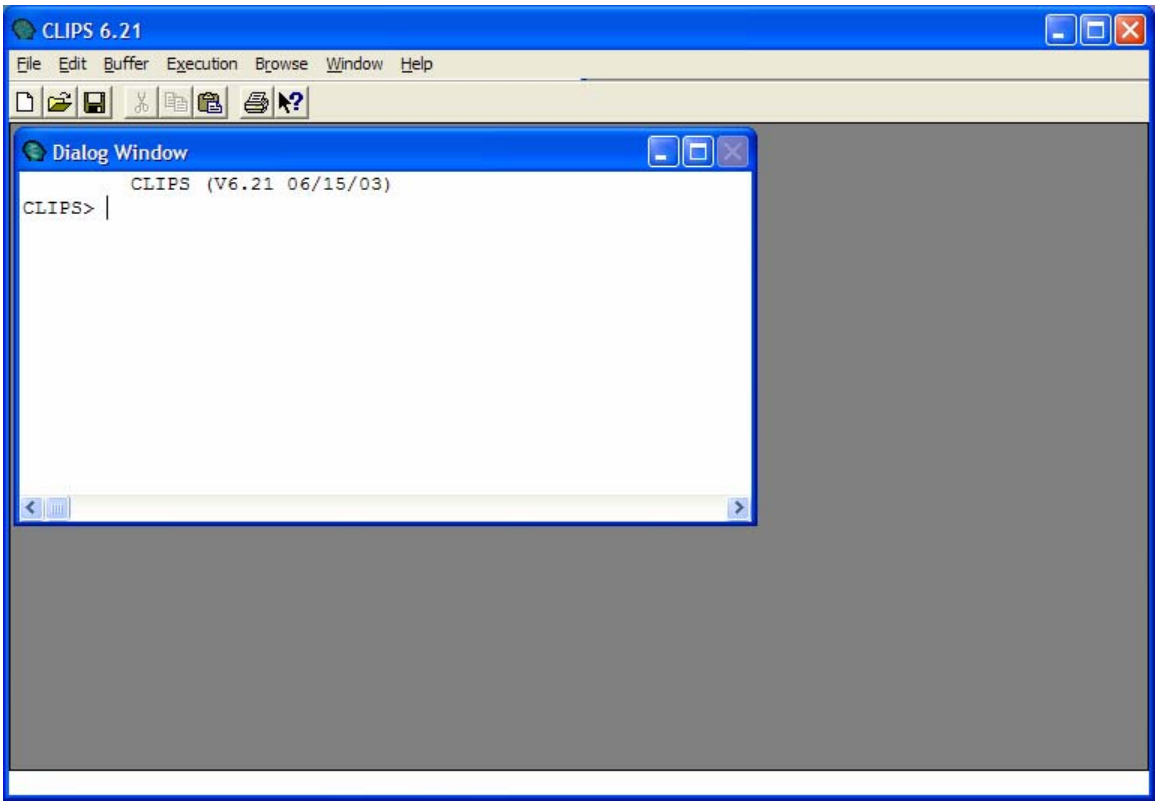


Figure E.1 Opening CLIPS window.

It is recommended to open the Agenda and Facts windows if the user whishes to see all the reasoning and inferences that goes on behind the scene. The Facts window shows all the facts that are created as the KBS program executes. The Agenda window shows

all the potential matches of rules with facts and the order in which these will be executed. These windows are opened by clicking on the "Window" button and selecting from the list Agenda and then Facts. For convenience these windows can be dragged and dropped in any place on the active desk top. See Figure E.2 as an example.

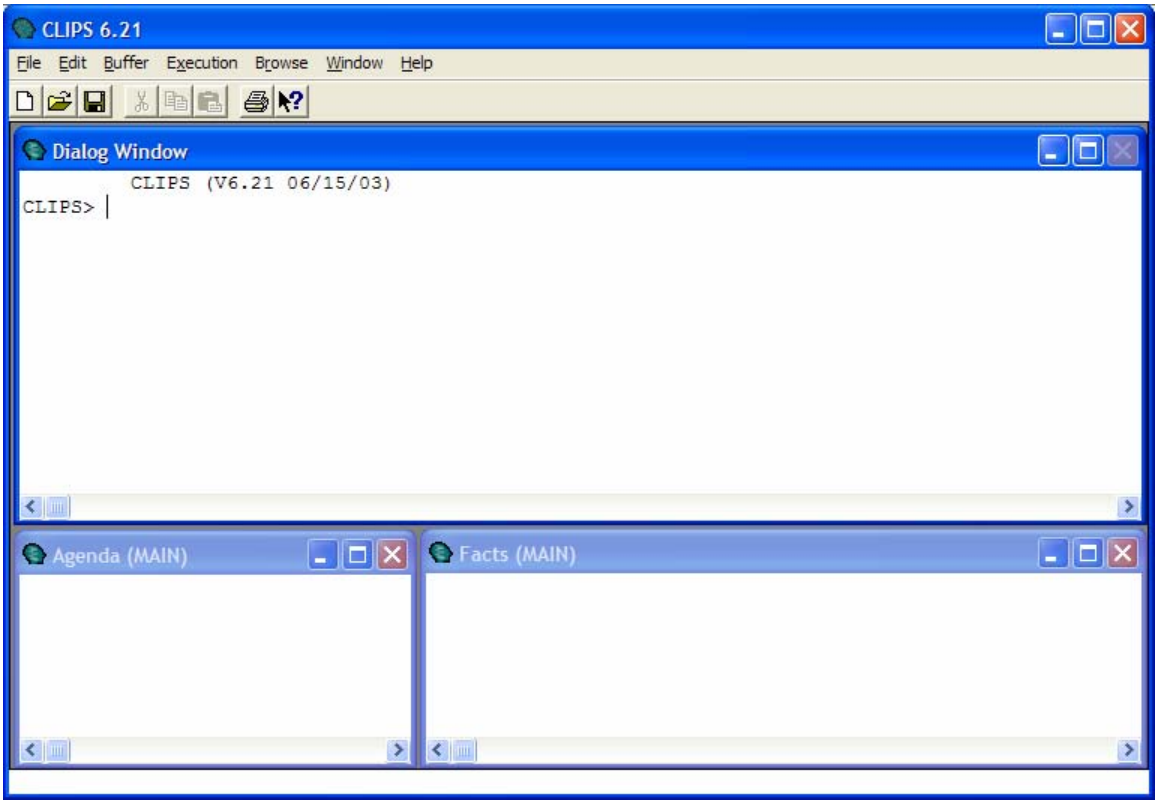


Figure E.2 Example of Agenda and Facts layout.

In order to load the KBS program follow these steps.

1. Click on "File".
2. Open the "Load Batch…" link.
3. Brows until the saved "pal.bat" file is found, and then click to open this file. The program will load at this point.

4. At the CLIPS> prompt, type in (reset). This will reset all the necessary facts. Note that 88 facts are loaded to begin with (Figure E.3). These facts can be viewed individually in the Facts (MAIN) window.

5. To execute the KBS merely type (run) at the CLIPS> prompt. (Figure E.4).

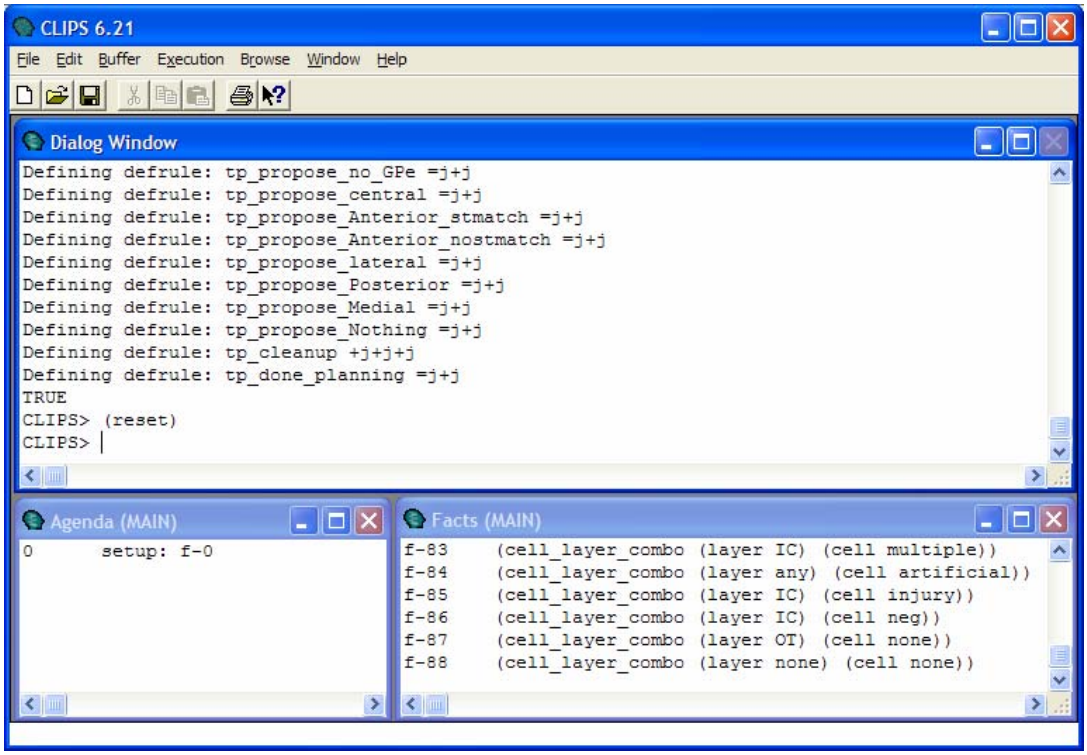


Figure E.3 Reset the KBS program.

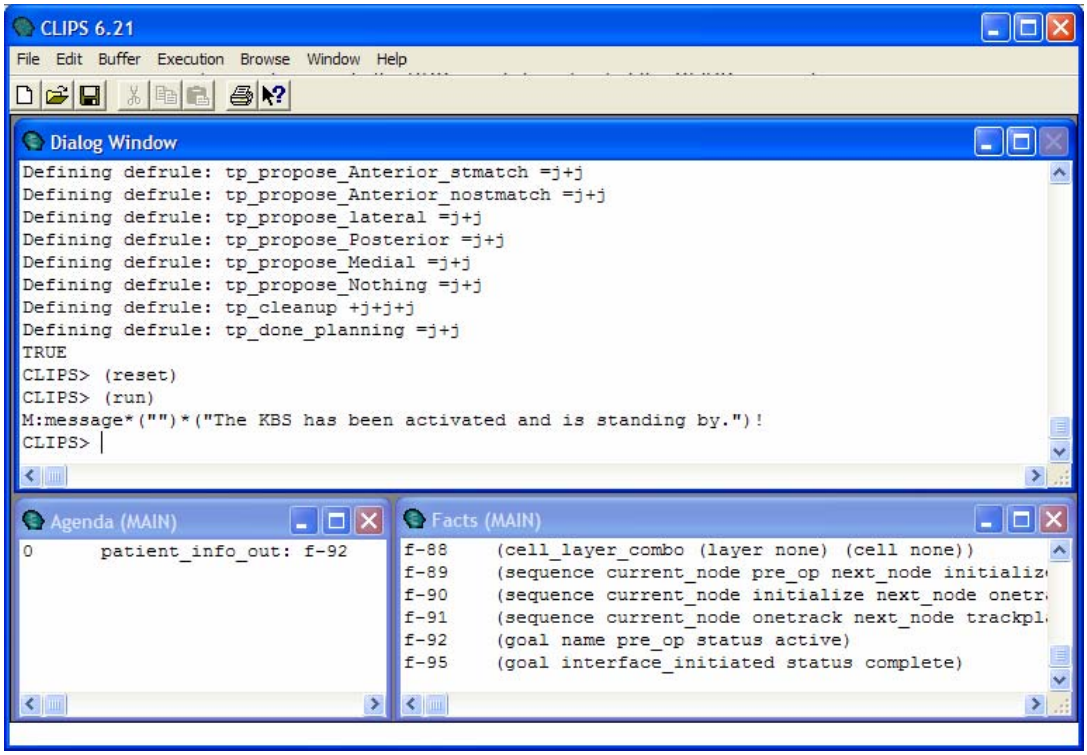


Figure E.4 KBS activated and ready to start.

E.4. KBS Language

The specific language construction is explained in more detail in Chapter 8. A test case example can be found in Appendix G. Here follows a brief description of the language parameters. There are two types of responses to the KBS, namely an answer response and an event response.

A. Answer Response

When prompted by the KBS to provide a set of information it is necessary to always assert the following statement. Assert means to type in a sentence at the CLIPS> prompt.

(assert (goal answer status available))

This assert notifies the KBS that an answer is available that needs to be processed. Following this there needs to be a second assert that contains the answer. The following is the generic format for that answer. Anything following a question mark is a variable that needs to be filled out by the user. Here is the format then,

(assert (answer (yn-answer ?yn)(multichoice ?mc)(timetag ?time)
(current-depth ?d)(text-field ?text)(pathplan ?path)(dsp-cell ?dsp)(user ?user)
(format ?form)(status ?stat)

For "?yn" the only acceptable answers are 'y' or 'n', indicating yes or no for the question asked. For "?mc" the acceptable answers are obtained from the list that are presented to the user for a decision. "?time" is a integer value that increases every instant a new assert is done to the KBS. The current depth value should always be given to the KBS during track execution and is placed in the "?d" slot. If the user wishes to make notes on any information being given to the KBS, then this can be done in the "?text" field. The "?path" has a set format that needs to be followed, see Figure E.5. Here is an example to illustrate how to enter the pathplan when prompted to do so

(assert (answer (timetag 3)(pathplan ST 43 Le 53.85 GPe 56.84 Li 63.61 GPi 64.43
La 68.21 OT 75.7 IC 76.92)(format model))).

The "?dsp", "?user", and "?d" slots should only be entered as an event response, which occurs during track execution. The "?form" slot typically indicates where the information is from or the format that the answer needs to be in. Typically the "?form" is specified by the KBS when it prompts the user for an answer.

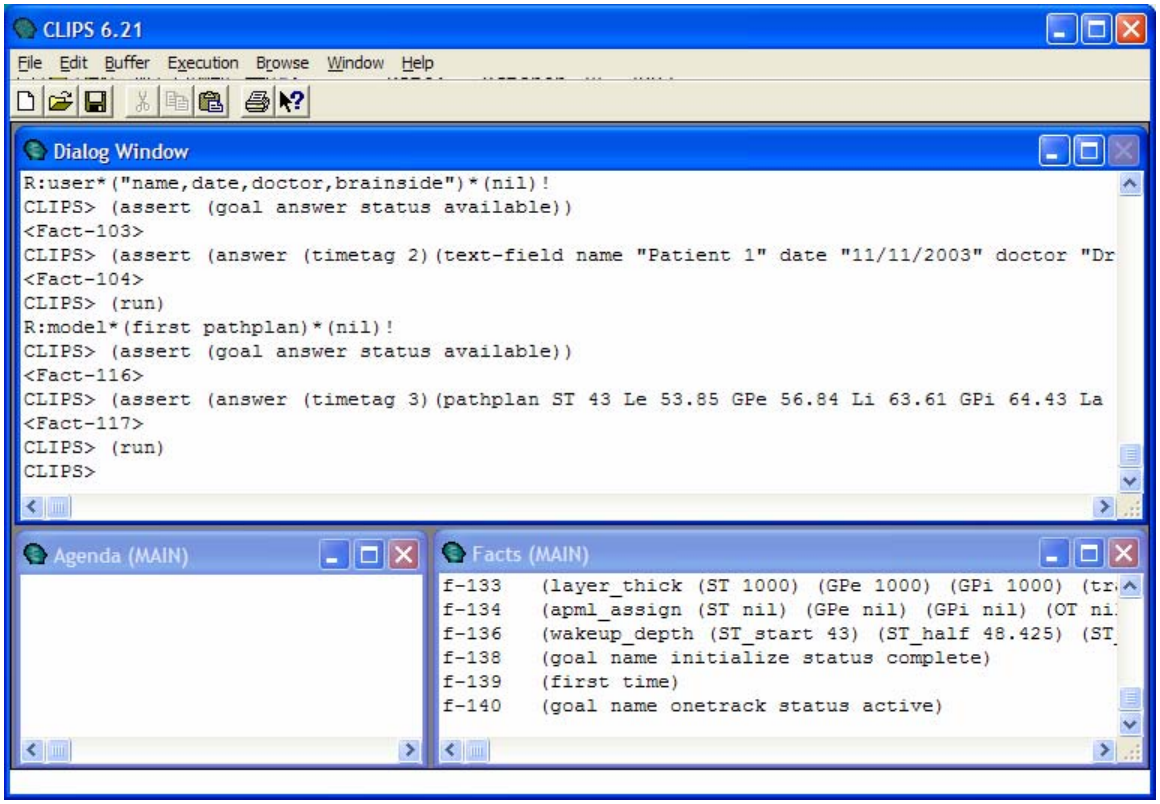


Figure E.5 Asserting the path plan

Note that it is necessary to (run) the program every time after the second assertion. This causes the KBS to activate and process the information that was made available to it. After the path plan is entered, the KBS enters the track execution phase. The KBS waits for event information to execute.

In the case where the user chooses to end the operation, the following assertion is made (the key is to have the status as endop):

(assert (goal answer status available))

(assert (answer (timetag 52)(status endop)))

No further reasoning will be done by the KBS even if more information is presented to it. Use this only when the operation is completed. All the facts will be stored in a

file titled “kbsdata.txt”. This allows the user to view all the decisions that were made during the course of the operation.

B. Event Response

To initiate KBS track execution it is necessary to provide a set of data to the KBS. As in Answer response an assert is done to notify the KBS that information is available. In a sense this wakes the KBS up. The assert is as follows:

(assert (goal name kbs status wakeup))

Note that this statement has to be asserted every time that an event is created by the user or the Onetrack system. As discussed in Chapter 7 there are three types of events.

1. Depth information is the only piece of information available. The assert for such an event is as follows:

   (assert (goal name kbs status wakeup))

   (assert (answer (timetag 6)(current-depth 52.85)(format depth)))

2. Depth and DSP information are available to the KBS. The assert for such an event is as follows:

   (assert (goal name kbs status wakeup))

   (assert (answer (timetag 17)(current-depth 62)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 50 LFD-B 20 border 30 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(format dsp)))

3. Depth, DSP and User information are available to the KBS. The assert for such an event is as follows:

(assert (goal name kbs status wakeup))

(assert (answer (timetag 20)(current-depth 63)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 40 HFD-P 20 LFD-B 60 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 20 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPe cell HFD-P)(format user)))

Figure E.6 is an example of when all three sources are made available to the KBS. Note the response from the KBS. First it tells the user the maximum depth he is allowed to go to without wakening the KBS. Secondly it gives the user a decision for the layer and cell type and specifies that this answer is solid, meaning it is absolutely known.

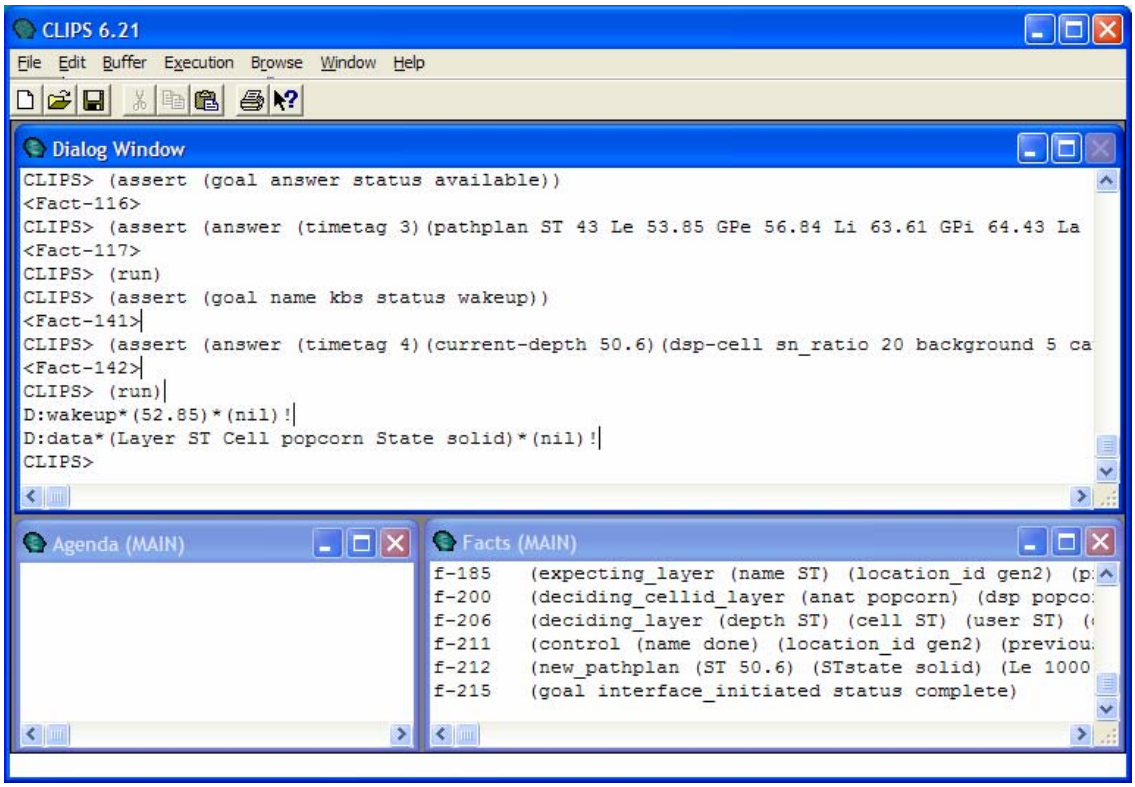


Figure E.6 Example when all input sources are available.

In the event that the user wants to complete the track, the following assert needs to be made:

(assert (goal answer status available))

(assert (answer (timetag 50)(status endtrack)))

The KBS will not allow new events to be made for a given track after that track is completed.

E.5. Conclusion

Running the KBS in the CLIPS program can be a very tedious process and should be done only to gain a greater understanding of the KBS program itself. The user interface through the Onetrack system takes care of all the syntax issues and makes it easier to deal with the KBS. When using CLIPS it is often beneficial to create an input file for the KBS, from which the user can merely copy and paste the responses into the KBS. An example of such a file is found in Appendix G. The best way to learn and understand the syntax involved with the KBS is by working through these test cases.

APPENDIX F

KBS Code Explanation

F.1 Summary

Number of Defglobal statements are 1.

Number of Deftemplate statements are 19.

Number of Deffact statements are 3.

Number of Defrule statements are 197.

Number of Deffunction statements are 5 .

F.2 Templates

Defglobal: Sets the devise to use as printout. This is to the CLIPS prompt and wdialog is to the C++ program.

Deftemplate answer: This template is used by the C++ to send information over to the CLIPS. An assert is done in this format to get information to the KBS.

Deftemplate anser_received: The input from C++ is put into this form for the KBS to process with, identical to *Deftemplate answer* however its function is for internal use in the KBS and not for interface purposes.

Deftemplate output: This is the format in which information is sent to the output rule. All info sent to the C++ contains these sets of information.

Deftemplate initial_info: Template set for initial patient information such as the date of operation.

Deftemplate anatomy_layer_down: Format to help state the order in which layers can occur as the probe moves downward and thus deeper into the brain.

Deftemplate anatomy_layer_up: Format to help state the order in which layers can occur as the probe moves upward and out of the brain.

Deftemplate pathplan: A pathplan is obtained from the 3D model of the Onetrack System. Important to note that the KBS translate this information to obtain the start and end depth for each layer.

Deftemplate control: This template is used to make sure that each piece of information in the KBS can be linked to previous information obtained during a track.

Deftemplate expecting_layer: Used internally by the depth reasoning. Based on the pathplan and current depth of probe a specific layer is expected to be found.

Deftemplate deciding_layer: This template will contain all the final reasoning decision on the probe layer type, done by depth, cell and user reasoning.

Deftemplate new_pathplan: At the completion of a track the KBS will produce a new proposed pathplan for the track.

Deftemplate cell_layer_combo: Certain cell types are associated with specific layers. This template will be the format in which to store that information.

Deftemplate deciding_cellid_layer: Different phases of the KBS are associated with cell type identification. This template will store the information of each phase for the cell type identification.

Deftemplate dsp_max: The DSP sends over probability array of cell types, the maximum three values will be stored in this format, for use in reasoning.

Deftemplate dsp_list: This is the template for the probability array of cell types.

Deftemplate user: This template is used to store information that the user inputs into the system.

Deftemplate wakeup_depth: The KBS sets a wakeup depth at which it must be awakened, if it has not be awakened before those points.

Deftemplate layer_thick: After track completion the new pathplan is analyzed. The thickness of each layer is calculated and stored in this template.

Deftemplate apml_assign: Depending on the thickness of each layer the user determines where in the apml (anterior posterior medial lateral) location of the brain the probe is located.

F.3 Facts

Deffacts downward_layering: Contains the combinations in which layers can be found as the probe goes deeper into the brain. These are the only combinations that are allowed during the KBS reasoning, and are a basic assumption.

Deffacts upward_layering: Contains the combinations in which layers can appear as the probe moves up relative to its current position.

Deffacts known_cell_layer_combinations: Certain cells are only associated with certain layers and this is also a strong basis in the KBS cell type reasoning.

F.4 Rules

TOPVIEW

Defrule setup: This is the order in which tasks must be done in Topview. The order is Pre Operations => Initialization => Onetrack

Defrule switch_rule: This is the mechanism used to switch between these primary tasks. When one task becomes complete the next one is activated.

Defrule done_goal_pre_op: All the requirements to ensure that the preop tasks are completed.

Defrule done_goal_initialize: All the requirements to ensure that the initialization tasks are done.

Defrule done_goal_onetrack: All the requirements to ensure that the onetrack tasks are done.

Defrule done_goal_trackplanning: All the requirements to ensure that the track planning tasks are done.

Defrule new_track: Activates the initialization tasks again.

Defrule done_goal_endop: Completes the operation.

PRE OPERATION

Defrule hello_out: This notifies the user that the KBS has been activated.

Defrule hello_in: This acknowledges from the user that the KBS is activated.

Defrule patient_info_out: This rule requests information from Onetrack concerning general patient information.

Defrule patient_info_in: This process the general patient information. It includes the patient name, date of operation, the name of the doctor performing the operation, and the side of the brain being operated on (left or right.)

INITIALIZE

Defrule request_pathplan: The KBS requests the pathplan from the Onetrack System.

Defrule path_plan_in: Takes the pathplan information and converts it into the right format that is used by the rest of the KBS.

Defrule initialize_templates: Initializes all of the templates that will be used during the execution of the track and track planning. Note the 1000 serve the same purpose as nil, it is a numerical nil.

Defrule set_depth_wakeup: The depths at which the KBS should be awakened is calculated here. Note no restriction placed that the KBS has to wake up at that exact depth. Rather it is a guideline. If no information is made available to the KBS before that depth, then it should be woken up.

ONE TRACK

Defrule determine_Infer_Layer: The Onetrack is a backward chaining process, and thus this rule establishes all the tasks that must be done before the KBS can determine what layer the probe is in. These tasks are *event wakeup, depth, cell* and *user*.

Defrule done_event_wakeup: All requirements to ensure that the event wakeup task is performed.

Defrule done_depth: All the requirements to ensure that the depth task is performed.

Defrule done_cell: All the requirements to ensure that the cell task is performed.

Defrule done_user: All the requirements to ensure that the user task is performed.

Defrule done_infer_layer: All the requirements to ensure that the layer determination tasks are done.

Defrule end_track: This rule when activated will cause the internal loop to exit. Thus a track execution will be complete.

EVENT WAKEUP

Defrule event_depth_only: Whenever only the current depth of the track is available this rule will gather the information and generate all the necessary facts to do the reasoning involved with depth only.

Defrule event_depth_dsp: When DSP probability array of cell types and the current depth of the probe is available for reasoning. All the necessary facts are generated to do the reasoning.

Defrule event_depth_user: When user cell type and layer type selection is available in addition to the DSP probability array of cell types and the current depth of the probe. All the necessary facts are generated to do the reasoning.

Defrule event_end_track: The input rule for when the user decides to end the track.

Defrule event_new_track: The input rule for when the user decides to start a new track.

Defrule event_end_op: The input rule for when the user decides to end the operation.

Defrule set_wakeup_STstart: Set the Striatum start wakeup depth. This happens when the current depth is before ST_start.

Defrule set_wakeup_SThalf: Set the Striatum half way wakeup depth. This happens when the current depth is after ST_start and before ST_half.

Defrule set_wakeup_STnend: Sets the Striatum end wakeup depth. This happens when the current depth is after ST_half but before ST_nend.

Defrule set_wakeup_Lehalf: Sets the Lamina Exterior wakeup depth. This happens when the current depth is after ST_nend but before Le_half.

Defrule set_wakeup_GPestart: Sets the GPe start wakeup depth. This happens when the current depth is after Le_half but before GPe_start.

Defrule set_wakeup_GPehalf: Sets the GPe half way depth wakeup. This happens when the current depth is after GPe_start but before GPe_half.

Defrule set_wakeup_GPenend: Sets the GPe end depth wakeup. This happens when the current depth is after GPe_half but before GPe_nend.

Defrule set_wakeup_Lihalf: Sets the Li half way depth wakeup. This happens when the current depth is after GPe_nend but before Li_half.

Defrule set_wakeup_GPistart: Sets the GPi start depth wakeup. This happens when the current depth is after Li_half but before GPi_start.

Defrule set_wakeup_GPihalf: Sets the GPi half way depth wakeup. This happens when the current depth is after GPi_start but before GPi_half.

Defrule set_wakeup_GPinend: Sets the GPi end depth wakeup. This happens when the current depths is after GPi_half but before GPi_nend.

Defrule set_wakeup_Lastart: Sets the La start depth wakeup. This happens when the current depth is after GP_nend but before La_start.

Defrule set_wakeup_La_OT_warning: Sets the La depth wakeup. This happens when the current depth is after La_start but before OT_start.

Defrule set_wakeup_La_IC_special: Sets the La start depth wakeup. This happens when the current depth is after La_start but before IC_start.

Defrule set_wakeup_OT_warning: Sets the OT wakeup. This sends a warning to the user that the probe is entering the Optic Track. Happens when the current depths it after the OT_start but before the IC_start.

Defrule set_wakeup_IC_warning: Sets the IC wakeup. This sends a warning to the user that the probe is entering the Internal Capsule. This happens when current depth is after the IC_start.

Defrule set_wakeup_nothing: These previous depth wakeup rules can only be fired once. If the situation exists again then this rule will fire making sure that the user and the Onetrack system does not get a repeated message.

DEPTH

Defrule determine_before_Striatum: Determines that from the pathplan that based on the current depth the probe is not yet in the Striatum.

Defrule Striatum_before: When the above happens this rule will assign none to the depth reasoning for layer.

Defrule determine_Striatum: Determines that from the pathplan that based on the current depth that the Striatum is the expected layer for the probe.

Defrule Striatum_downward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found if the probe moved down. Expected layer must be ST.

Defrule Striatum_upward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found in the probe moved up. Expected layer must be ST.

Defrule first_time: A special case exists if this is the first time that the depth is being compared; the answer is not compared but taken at face value.

Defrule determine_Lamina_exterior: Determines that from the pathplan that based on the current depth that the LE is the expected layer for the probe.

Defrule Le_downward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found in the probe moved down. Expected layer must be Le.

Defrule Le_upward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found if the probe moved up. Expected layer must be Le.

Defrule determine_GPe: Determines that from the pathplan that based on the current depth that the GPe is the expected layer for the probe.

Defrule GPe_downward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found in the probe moved down. Expected layer must be GPe.

Defrule GPe_upward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found if the probe moved up. Expected layer must be GPe.

Defrule determine_Li: Determines that from the pathplan that based on the current depth that the Li is the expected layer for the probe.

Defrule Li_downward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found in the probe moved down. Expected layer must be Li.

Defrule Li_upward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found if the probe moved up. Expected layer must be Li.

Defrule determine_GPi: Determines that from the pathplan that based on the current depth that the GPi is the expected layer for the probe.

Defrule GPi_downward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found in the probe moved down. Expected layer must be GPi.

Defrule GPi_upward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found if the probe moved up. Expected layer must be GPi.

Defrule determine_La_OT: Determine that from the pathplan that based on the current depth that the La is the expected layer for the probe. This happens if the OT depth is available in the pathplan.

Defrule determine_La_IC: Determine that from the pathplan that based on the current depth that the La is the expected layer for the probe. This happens when the IC depth is available in the pathplan.

Defrule La_downard: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found if the probe moves down. Expected layer must be La.

Defrule La_upward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found if the probe moved up. Expected layer must be La.

Defrule determine_OT: Determines that from the pathplan that based on the current depth that the OT is the expected layer for the probe.

Defrule OT_downward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found in the probe moved down. Expected layer must be OT.

Defrule OT_upward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found if the probe moved up. Expected layer must be OT.

Defrule determine_IC: Determines that from the pathplan that based on the current depth that the IC is the expected layer for the probe.

Defrule IC_downward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found in the probe moved down. Expected layer must be IC.

Defrule IC_upward: Determines if the layer combinations are correct and that the layer selection makes sense based on the previous layer found if the probe moved up. Expected layer must be IC.

CELL ANATOMY

Defrule cell_expected_done: When the assignment of depth selection for cell type has been completed this rule will complete the goal.

Defrule cell_expected_none: In the case where the expected layer is *none* this rule will assign *none* to cell type selection for anatomy.

Defrule cell_expected_ST: In the case where the expected layer is *ST* this rule will assign *popcorn, border,* and *caudate* to cell type selection for anatomy.

Defrule cell_expected_Le: In the case where the expected layer is *Le* this rule will assign *fiber* and *border* to the cell type selection for anatomy.

Defrule cell_expected_GPe: In the case where the expected layer is *GPe* this rule will assign *HFD-P, LFD-B,* and *border* to the cell type selection for anatomy.

Defrule cell_expected_Li: In the case where the expected layer is *Li* this rule will assign *fiber* and *border* to the cell type selection for anatomy.

Defrule cell_expected_GPi: In the case where the expected layer is *GPi* this rule will assign *tonic, bursting, chugging, pausing, tremor* and *border* to the cell type selection for anatomy.

Defrule cell_expected_La: In the case where the expected layer is *La* this rule will assign *fiber* and *border* to the cell type selection for anatomy.

Defrule cell_expected_IC: In the case where the expected layer is *IC* this rule will assign *multiple, injury* and *negative* to the cell type selection for anatomy.

Defrule cell_expected_OT: In the case where the expected layer is *OT* this rule will assign *none* to the cell type selection for anatomy.

DSP

Defrule sn_primary_structure: If the signal/noise ratio is greater than 15 then this rule will reason that the probe must be seeing the cell type of a primary layer (ST, GPe or GPi).

Defrule sn_secondary_structure: If the signal/nose ratio is less than 15 then this rule will reason that the probe must be seeing the cell type of a secondary layer (Le, Li, or La).

Deffunction find_max: This function finds the maximum value in a list that is associated with the list of cell type probabilities.

Deffunction replace: This is a replacement function. It normalizes the previous selected cell type and zeroing it out, so that when the second max is found it does not refer to the first max.

Deffunction cellorder_maxval: Assigns the maxvalue label to the dsp_max template. For example, it will assert popcorn as oppose the value associated with popcorn.

Deffunction cellorder_2maxval: Assigns the 2nd max value label to the dsp_max template.

Deffunction cellorder_3maxval: Assigns the 3rd max value label to the dsp_max template.

Defrule find_max_cell: This rule is responsible for passing info to the deffunctions in order to determine the maximum value in the cell type probability list.

Defrule find_2max_cell: This rule is responsible for passing info to the deffunctions in order to determine the 2nd maximum value in the cell type probability list.

Defrule find_3max_cell: This rule is responsible for passing info to the deffunctions in order to determine the 3rd maximum value in the cell type probability list.

Defrule dsp_primary_max1: If the primary layer type is assigned, then the combination for layer->cell type will be compared, to ensure that maximum 1 corresponds. This rule assigns the final dsp cell type answer.

Defrule dsp_primary_max2: If the primary layer type is assigned, then the combination for layer-> cell type will be compared to the maximum 2 answer. This rule assigns the final dsp cell type answer.

Defrule dsp_primary_max3: If the primary layer type is assigned, then the combination for layer -> cell type will be compared to the maximum 3 answer. This rule assigns the final dsp cell type answer.

Defrule dsp_noprimary_max1: When it is a primary layer but the Max 1 does not correspond to the layer/cell type combo, and it is not an artificial sound. Then it will assign *none* as the final dsp cell type answer.

Defrule dsp_specrimary_max1: When it is a primary layer but Max 1 is an artificial sound, send a warning message to the user and assign *none* as the final dsp cell type answer.

Defrule dsp_noprimary_max2: When it is a primary layer but the Max 2 does not correspond to the layer/cell type combo then assign *none* as the final dsp cell type answer.

Defrule dsp_noprimary_max3: When it is a primary layer but the Max 3 does not correspond to the layer/cell type combo then assign *none* as the final dsp cell type answer.

Defrule dsp_primary_done: When it is a primary layer and all the maxes (1,2,3) have been processed this rule will fire to complete this assignment.

Defrule dsp_secondary_max1: If the secondary layer type is assigned, then the combination for layer->cell type will be compared, to ensure that maximum 1 corresponds. This rule assigns the final dsp cell type answer.

Defrule dsp_secondary_max2: If the secondary layer type is assigned, then the combination for layer/cell type will be compared to ensure that maximum 2 corresponds. This rule assigns the final dsp cell type answer.

Defrule dsp_secondary_max3: If the secondary layer type is assigned, then the combination for layer/cell type will be compared to ensure the maximum 3 corresponds. This rule assigns the final dsp cell type answer.

Defrule dsp_nosecondary_max1: When it is a secondary layer but Max 1 corresponds to the primary layer cell types, is not a border or artificial then assign *none* as the dsp cell type answer.

Defrule dsp_specsecondary_max1: When it is a secondary layer and Max 1 is artificial then notify user to check equipment and assign *none* to the dsp cell type answer.

Defrule dsp_nosecondary_max2: When it is a secondary layer but Max 2 corresponds to the primary layer cell types, is not a border then assign *none* as the dsp cell type answer.

Defrule dsp_nosecondary_max3: When it is a secondary layer but Max 3 corresponds to the primary layer cell type, is not a border then assign *none* as the dsp cell type answer.

Defrule dsp_secondary_done: After completing all the secondary max assignments this rule will complete the task.

Defrule no_dsp: In the case where there is no dsp input this rule will fire, and will allow reasoning to pass straight through this task.

USER

Defrule user_motor: This is a holder place that will allow future input from the user regarding sensory motor inputs.

Defrule user_cell: Assigns the user selection to the cell type answer.

Defrule user_cell_done: Makes sure that all templates are filled in with the users answer to the cell type.

Defrule user_cell_none: When there is no user input this rule will assign *none* to the users answer to the cell type.

Defrule user_layer: Assigns the user selection to the layer type answer.

Defrule user_layer_none: When there is no user input this rule will assign *none* to the user answer to the layer type.

Defrule user_done_input: When all user tasks have been done, this rule will complete this task.

COMPARE CELL TYPE

Defrule compare_cell_cunod_cumatch: When anatomy is *none* and dsp and user answers match, then assign the match as the final cell type choice.

Defrule compare_cell_ainput_nomatch: When anatomy is the only answer and dsp and user is *none*, and then assign *none* as the final cell type choice because you can not base the cell type selection off of depth reasoning alone.

Defrule compare_cell_cunod_nomatch: When anatomy is *none* and dsp and user does not match, then assign *user* answer as the final cell type choice because user has higher validity.

Defrule compare_cell_allinput_allmatch: When anatomy, dsp and user match assign the match as the final cell type choice.

Defrule compare_cell_allinput_admatch: All input available, anatomy and dsp match, but does not match with user. Warn user of possible mistake and ask them to re-enter cell type.

Defrule compare_cell_verify: Takes the user answer from the previous defrule and assigns it as the final cell type choice.

Defrule compare_cell_allinput_aumatch: All input available, anatomy and user match, but does not match with dsp. Assign users answer as the final cell type choice.

Defrule compare_cell_allinput_dumatch: All input available, dsp and user match, but does not match with anatomy. Assign users answer as the final cell type choice.

Defrule compare_cell_allinput_nomatch: All input available, but nothing matches. Assign the users answer as the final cell type choice.

Defrule compare_cell_adinput_allmatch: The anatomy and dsp is available, and the match. Also the dsp value must be max 1. Assign the match as the final cell type choice.

Defrule compare_cell_adinput_halfmatch: The anatomy and dsp is available and matches. However the dsp value is not the max 1. then assign *none* as the final cell type choice.

Defrule compare_cell_adinput_nomatch: The anatomy and dsp is available but do not match. Assign *none* as the final cell type choice.

Defrule compare_cell_cleanup: Retract any cell id layers that have not been reasoned with.

Defrule compare_cell_done: When cleanup is done complete the comparison task.

Defrule cell_to_layer: Once the final cell type choice has been made this rule will assign the corresponding layer to the cell type.

INFER LAYER

Defrule determine_layer_dcusame: The depth, dsp and user selection for layer type matches. Assign the matched layer as the final layer type.

Defrule determine_layer_udsame: The depth and user selection for layer type matches, but not cell. Assign the match as the final layer type.

Defrule determine_layer_ucsame: The dsp and user selection for layer type matches, but not the depth. Assign the match as the final layer type.

Defrule determine_layer_cdsame: The depth and dsp selection for layer type matches, but not the user. Notify user of potential mistake, and ask for them to re-input their layer type selection.

Defrule determine_layer_verify: Take the input from the previous rule and assign it as the final layer type.

Defrule determine_layer_dcunotsame: The depth, dsp and user selection for layer type does not match. Assign the user selection as the final layer type.

Defrule determine_layer_depthonly: If depth is only answer available, then assign the depth answer as the final layer type.

Defrule determine_layer_dconly: If depth and cell type is the only answer and they match then assign the match answer as the final layer type.

Defrule determine_layer_dconlyno: If depth and cell type is the only answer and does not match then assign the cell answer as the final layer type.

Defrule determine_layer_cunod: If only the cell and user layer decisions are available then choose the user answer as the final layer type.

Defrule layertype_same: If the current assigned layer type is the same as the previous one then do not update the found path plan.

Defrule update_path_first: A special case exists for the first time. This rule makes sure that the first time the KBS does this it fires the right rules.

Defrule layertype_change: When the current assigned layer is not the same as the previous layer then update the found path plan.

Defrule update_path_none: If the current assigned layer is *none,* then do not update the found path plan.

Defrule update_path_ST: If the current assigned layer is *ST* then update the path plan. However this can only be done once for a given track.

Defrule update_path_ST_skip: If the current assigned layer is *ST*, but the path plan has already been updated for the ST start then skip the found path plan assignment.

Defrule update_path_Le: If the current assigned layer is *Le* then update the path plan. However this can only be done once for a given track.

Defrule update_path_Le_skip: If the current assigned layer is *Le*, but the path plan has already been updated for the Le start then skip the found path plan assignment.

Defrule update_path_GPe: If the current assigned layer is *GPe* then update the path plan. However this can only be done once for a given track.

Defrule update_path_GPe_skip: If the current assigned layer is *GPe*, but the path plan has already been updated for the GPe start then skip the found path plan assignment.

Defrule update_path_Li: If the current assigned layer is *Li* then update the path plan. However this can only be done once for a given track.

Defrule update_path_Li_skip: If the current assigned layer is *Li*, but the path plan has already been updated for the Li start then skip the found path plan assignment.

Defrule update_path_GPi: If the current assigned layer is *GPi* then update the path plan. However this can only be done once for a given track.

Defrule update_path_GPi_skip: If the current assigned layer is *GPi*, but the path plan has already been updated for the GPi start then skip the found path plan assignment.

Defrule update_path_La: If the current assigned layer is *La* then update the path plan. However this can only be done once for a given track.

Defrule update_path_La_skip: If the current assigned layer is *La*, but the path plan has already been updated for the GPi start then skip the found path plan assignment.

Defrule update_path_OT: If the current assigned layer is *OT* then update the path plan. However this can only be done once for a given track.

Defrule update_path_OT_skip: If the current assigned layer is *OT*, but the path plan has already been updated for the OT start then skip the found path plan assignment.

Defrule update_path_IC: If the current assigned layer is *IC* then update the path plan. However this can only be done once for a given track.

Defrule update_path_IC_skip: If the current assigned layer is *IC*, but the path plan has already been updated for the IC start then skip the found path plan assignment.

TRACK PLANNING

Defrule cleanup_track: After the onetrack loop is completed, certain facts will need to be retracted in order to have a successful second track execute. This rule gets rid of all the unwanted facts.

Defrule tp_calculate_layer_thickness: This rule calculates the thickness of each primary layer as determined by the new path plan.

Defrule tp_ST_anterior_lateral: If the ST thickness is greater than 8 then the probe can be anterior or lateral.

Defrule tp_ST_posterior_medial: If the ST thickness is less than or equal 8 then the probe can be posterior or medial.

Defrule tp_ST_none: If no ST was found then this rule will assign *none* for the apml orientation.

Defrule tp_GPe_anterior: If the GPe thickness is greater than 6 then the probe can be anterior.

Defrule tp_GPe_central: If the GPe thickness is less than or equal to 6 and greater than or equal to 4 then the probe can be central.

Defrule tp_GPe_none: If no GPe was found then this rule will assign *none* for the apml orientation.

Defrule tp_GPi_apl: If the GPi thickness is less than or equal to 4 then the probe can be anterior, posterior or lateral.

Defrule tp_GPi_medial: If the GPi thickness is greater than 4 and less than 6 than the probe can be medial.

Defrule tp_GPi_central: If the GPi is greater than or equal to 6 then the probe is central.

Defrule tp_GPi_none: If no GPi was found then this rule will assign *none* for the apml orientation.

Defrule tp_OT_positive: If OT is found then the probe has a positive OT assignment.

Defrule tp_OT_negative: If OT is not found then the probe has a negative OT assignment.

Defrule tp_IC_positive: If IC is found then the probe has a positive IC assignment.

Defrule tp_IC_negative: If IC is not found then the probe has a negative IC assignment.

Defrule tp_done_apml_assign: When all the possible combinations have been assigned then this task is completed. Activate the planning where next track should be placed task.

Defrule tp_propose_no_GPi: If there is no GPi then the user must be notified, and the KBS can not suggest where the next track should go.

Defrule tp_propose_in: When a message is sent, the Onetrack system must acknowledge that the message was received. This rules receives that acknowledgment and completes the reasoning.

Defrule tp_propose_IC_pos: When IC is found the probe should move 2mm medial or lateral. Message sent to user.

Defrule tp_propose_OTpos_ICpos: If both OT and IC is present, note and move on to find just IC. Thus move posterior is small increments. Message sent to user. Message sent to user.

Defrule tp_propose_OTpos_ICneg: If OT is found the probe should move 2mm posterior.

Defrule tp_propose_no_ST: If no ST is found the user must check equipment and the KBS can not suggest where the probe should move to. Message sent to user. IC and OT negative.

Defrule tp_propose_no_GPe: If no GPe is found the user must check equipment and the KBS can not suggest where to probe should move to. Message sent to user. IC and OT negative.

Defrule tp_propose_central: If GPe and GPi suggests central then move the probe 2mm posterior until you find the IC. Message sent to user. IC and OT negative.

Defrule tp_propose_Anterior_stmatch: If ST, GPe and GPi suggests anterior then move the probe 3-4 mm posterior. Message sent to user. IC and OT negative.

Defrule tp_propose_Anterior_nostmatch: If GPe and GPi suggest anterior, but ST does not, still move the probe 3-4 mm posterior. Message sent to user. IC and OT negative.

Defrule tp_propose_lateral: If the ST and GPi suggests lateral, then move the probe anterior, medial or posterior. Message sent to user. IC and OT negative.

Defrule tp_propose_Posterior: If the ST and GPi suggests posterior, then move the probe 1mm posterior to find IC. Message sent to user. IC and OT negative.

Defrule tp_propose_Medial: If the ST and GPi suggests medial, then move the probe 2mm lateral or complete mapping. Message sent to user. IC and OT negative.

Defrule tp_propose_Nothing: If ST, GPe and GPi do not match the KBS is unable to suggest where the probe should be moved to. Message sent to user.

IC and OT negative.

Defrule tp_cleanup: Cleanup of unnecessary facts needs to be done.

Defrule tp_done_planning: When the cleanup is completed then this task is done.

INTERFACE

Defrule interfae_KBS_out: Whenever output is send to the Onetrack system it should pass through here. Note this rule halts, which means that the Onetrack system reads the information immediately and responds to it.

Defrule set_wakeup_out: Special output rule to set up the wakeup depths. Note that the only difference is that this rule does not halt the KBS.

Defrule interfae_KBS_in: Deals with generic input from Onetrack system.

Defrule KBS_wakeup_in: This rule deals specifically with info sent to the KBS when an event occurs. This can only work during the Onetrack loop.

Defrule end_track_input: Whenever the user decides to end the track, that info is received in this rule.

Defrule end_operation_input: Whenever the user decides to end the operation, that info is received in this rule.

APPENDIX G

KBS Testing in CLIPS Environment

G.1. Generic KBS Input File.

```
;REQUEST PRE OP
 (assert (goal answer status available))
(assert (answer (timetag 2)(text-field name "Linda Harley" date "Feb 8th, 2003" doctor "Dr. Klause" brainside "left")(format user)))

;PATHPLAN
(assert (goal answer status available) )
(assert (answer (timetag 3)(pathplan ST 50 Le 59 GPe 61 Li 67 GPi 69 La 78 OT 1000 IC 82)(format model)))

;Find ST
(assert (goal name kbs status wakeup))
(assert (answer (timetag 4)(current-depth 50.2)(format depth)))

;Find Le
(assert (goal name kbs status wakeup))
(assert (answer (timetag 5)(current-depth 59.6)(format depth)))

;Find GPe
(assert (goal name kbs status wakeup))
(assert (answer (timetag 6)(current-depth 61.2)(format depth)))
```

;Find Li

(assert (goal name kbs status wakeup))

(assert (answer (timetag 7)(current-depth 67.4)(format depth)))

;Find GPi

(assert (goal name kbs status wakeup))

(assert (answer (timetag 8)(current-depth 70)(format depth)))

(assert (goal name kbs status wakeup))

(assert (answer (timetag 9)(current-depth 73)(format depth)))

;Find La

(assert (goal name kbs status wakeup))

(assert (answer (timetag 10)(current-depth 78.6)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 30 HFD-P 60 LFD-B 50 border 30 tonic 30 bursting 30 chugging 30 pausing 30 fiber 90 multiple 30 artificial 30 injury 30 tremor 30 neg 10)(user layer La cell fiber)(format user)))

;Find OT

(assert (goal name kbs status wakeup))

(assert (answer (timetag 11)(current-depth 81)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 30 HFD-P 60 LFD-B 50 border 30 tonic 30 bursting 30 chugging 30 pausing 30 fiber 30 multiple 30 artificial 30 injury 30 tremor 30 neg 90)(user layer OT cell neg)(format user)))

;Find IC

(assert (goal name kbs status wakeup))

(assert (answer (timetag 11)(current-depth 83)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 30 HFD-P 60 LFD-B 50 border 30 tonic 30 bursting 30 chugging 30 pausing 30 fiber 30 multiple 30 artificial 30 injury 30 tremor 30 neg 90)(user layer IC cell neg)(format user)))

(assert (goal name kbs status wakeup))

(assert (answer (timetag 12)(current-depth 84)(format depth)))

;END track

(assert (goal answer status available))

(assert (answer (timetag 13)(status endtrack)))

;START new track

(assert (answer (timetag 14)(format tp)))

(assert (answer (timetag 15)(status newtrack)))

****

;PATHPLAN

(assert (goal answer status available))

(assert (answer (timetag 16)(pathplan ST 51 Le 59 GPe 61 Li 67 GPi 69 La 78 OT 1000 IC 82)(format model)))

;Find ST

(assert (goal name kbs status wakeup))

(assert (answer (timetag 17)(current-depth 50.2)(format depth)))

(assert (goal name kbs status wakeup))

(assert (answer (timetag 18)(current-depth 55)(format depth)))

;Find Le

(assert (goal name kbs status wakeup))

(assert (answer (timetag 19)(current-depth 59.6)(format depth)))

;Find GPe

(assert (goal name kbs status wakeup))

(assert (answer (timetag 20)(current-depth 61.2)(format depth)))

;Find Li

(assert (goal name kbs status wakeup))

(assert (answer (timetag 21)(current-depth 67.4)(format depth)))

;Find GPi

(assert (goal name kbs status wakeup))

(assert (answer (timetag 22)(current-depth 70)(format depth)))

(assert (goal name kbs status wakeup))

(assert (answer (timetag 23)(current-depth 73)(format depth)))

;Find La

(assert (goal name kbs status wakeup))

(assert (answer (timetag 23)(current-depth 78.6)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 30 HFD-P 60 LFD-B 50 border 30 tonic 30 bursting 30 chugging 30

pausing 30 fiber 90 multiple 30 artificial 30 injury 30 tremor 30 neg 10)(user layer La cell fiber)(format user)))

;Find OT

(assert (goal name kbs status wakeup))

(assert (answer (timetag 24)(current-depth 81)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 30 HFD-P 60 LFD-B 50 border 30 tonic 30 bursting 30 chugging 30 pausing 30 fiber 30 multiple 30 artificial 30 injury 30 tremor 30 neg 90)(user layer OT cell neg)(format user)))

;Find IC

(assert (goal name kbs status wakeup))

(assert (answer (timetag 24)(current-depth 83)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 30 HFD-P 60 LFD-B 50 border 30 tonic 30 bursting 30 chugging 30 pausing 30 fiber 30 multiple 30 artificial 30 injury 30 tremor 30 neg 90)(user layer IC cell neg)(format user)))

(assert (goal name kbs status wakeup))

(assert (answer (timetag25)(current-depth 83)(format depth)))

;END track

(assert (goal answer status available))

(assert (answer (timetag 26)(status endtrack)))

;END operation

(assert (answer (timetag 27)(format tp)))

(assert (answer (timetag 28)(status endop)))

G.2 KBS Input File

Patient 1

```
;       ---------------------------------------------------------------------------------------------
;                               TEST CASE 1 : INPUT
;       Author: Linda Harley
;       Date:   October 28, 2003
;       File:   test1input.txt
;       Updates:
;       ---------------------------------------------------------------------------------------------
-1-
(assert (goal answer status available))
(assert (answer (timetag 1)(format message)(status received)))
(run)
-2-
(assert (goal answer status available))
(assert (answer (timetag 2)(text-field name "Patient 1" date "11/11/2003" doctor "Dr. Y"
brainside "left")(format user)))
(run)
-3-
(assert (goal answer status available))
(assert (answer (timetag 3)(pathplan ST 43 Le 53.85 GPe 56.84 Li 63.61 GPi 64.43 La
68.21 OT 75.7 IC 76.92)(format model)))
(run)
```

-4-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 4)(current-depth 50.6)(dsp-cell sn_ratio 20 background 5 caudate 40 popcorn 60 HFD-P 20 LFD-B 20 border 20 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer ST cell none)(format user)))

(run)

-5-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 5)(current-depth 51.4)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer ST cell none)(format user)))

(run)

-6-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 6)(current-depth 52.85)(format depth)))

(run)

-7-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 7)(current-depth 55.345)(format depth)))

(run)

-8-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 8)(current-depth 56)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 60 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer ST cell none)(format user)))

(run)

-9-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 9)(current-depth 56.84)(format depth)))

(run)

-10-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 10)(current-depth 57.2)(dsp-cell sn_ratio 20 background 5 caudate 80 popcorn 60 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPe cell none)(format user)))

(run)

-11-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 11)(current-depth 60.2)(dsp-cell sn_ratio 20 background 5 caudate 50 popcorn 60 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer ST cell popcorn)(format user)))

(run)

(assert (goal answer status available))

(assert (answer (timetag 12)(multichoice popcorn)(format MC)))

(run)

-12-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 13)(current-depth 60.225)(format depth)))

(run)

-13-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 14)(current-depth 60.3)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer ST cell popcorn)(format user)))

(run)

(assert (goal answer status available))

(assert (answer (timetag 15)(multichoice popcorn)(format MC)))

(run)

-14-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 16)(current-depth 61.8)(dsp-cell sn_ratio 10 background 5 caudate 20 popcorn 30 HFD-P 10 LFD-B 10 border 40 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer Le cell border)(format user)))

(run)

-15-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 17)(current-depth 62)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 50 LFD-B 20 border 30 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(format dsp)))

(run)

-16-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 18)(current-depth 62.3)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 40 HFD-P 20 LFD-B 60 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(format dsp)))

(run)

-17-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 19)(current-depth 62.61)(format depth)))

(run)

-18-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 20)(current-depth 63)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 40 HFD-P 20 LFD-B 60 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 20 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPe cell HFD-P)(format user)))

(run)

-19-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 21)(current-depth 63.3)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 60 LFD-B 50 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPe cell LFD-B)(format user)))

(run)

-20-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 22)(current-depth 63.9)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 50 tremor 10 neg 10)(format dsp)))

(run)

-21-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 23)(current-depth 63.9)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 60 LFD-B 60 border 62 tonic 61 bursting 60 chugging 40 pausing 30 fiber 20 multiple 38 artificial 23 injury 20 tremor 18 neg 32)(user layer GPe cell HFD-P)(format user)))

(run)

(assert (goal answer status available))

(assert (answer (timetag 24)(multichoice HFD-P)(format MC)))

(run)

-22-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 26)(current-depth 64.10)(format depth)))

(run)

-23-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 27)(current-depth 64.2)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 50 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPe cell border)(format user)))

(run)

-24-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 28)(current-depth 64.46)(format depth)))

(run)

-25-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 29)(current-depth 65.1)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPe cell HFD-P)(format user)))

(run)

-26-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 30)(current-depth 65.3)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 50 LFD-B 10 border 10 tonic 10 bursting 10 chugging 20 pausing 50 fiber 10 multiple 10 artificial 10 injury 10 tremor 30 neg 10)(user layer GPe cell HFD-P)(format user)))

(run)

(assert (goal answer status available))

(assert (answer (timetag 31)(multichoice HFD-P)(format MC)))

(run)

-27-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 32)(current-depth 66)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 30 LFD-B 20 border 60 tonic 20 bursting 20 chugging 28 pausing 24 fiber 23 multiple 23 artificial 20 injury 10 tremor 10 neg 10)(user layer Li cell border)(format user)))

(run)

-28-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 33)(current-depth 66.4)(format depth)))

(run)

-29-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 34)(current-depth 67.3)(dsp-cell sn_ratio 10 background 4 caudate 11 popcorn 11 HFD-P 11 LFD-B 20 border 60 tonic 20 bursting 22 chugging 22 pausing 23 fiber 23 multiple 23 artificial 22 injury 20 tremor 10 neg 10)(user layer Li cell border)(format user)))

(run)

-30-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 35)(current-depth 68)(dsp-cell sn_ratio 20 background 5 caudate 22 popcorn 22 HFD-P 30 LFD-B 11 border 23 tonic 50 bursting 21 chugging 43 pausing 23 fiber 22 multiple 22 artificial 22 injury 20 tremor 10 neg 10)(user layer GPi cell tonic)(format user)))

(run)

-31-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 36)(current-depth 68.28)(format depth)))

(run)

-32-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 37)(current-depth 68.3)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 50 bursting 20 chugging 10 pausing 50 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPi cell tonic)(format user)))

(run)

-33-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 38)(current-depth 68.5)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 30 bursting 30 chugging 30 pausing 30 fiber 20 multiple 20 artificial 20 injury 20 tremor 20 neg 20)(user layer GPi cell pausing)(format user)))

(run)

-34-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 39)(current-depth 69.4)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPi cell pausing)(format user)))

(run)

-35-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 40)(current-depth 69.6)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 40 LFD-B 30 border 10 tonic 10 bursting 40 chugging 30 pausing 30 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPi cell bursting)(format user)))

(run)

-36-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 41)(current-depth 70.5)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 40 LFD-B 40 border 60 tonic 20 bursting 50 chugging 40 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPi cell border)(format user)))

(run)

(assert (goal answer status available))

(assert (answer (timetag 42)(multichoice GPi)(format MC)))

(run)

-37-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 43)(current-depth 70.9)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 30 bursting 60 chugging 50 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPi cell chugging)(format user)))

(run)

-38-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 44)(current-depth 71)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 30 tonic 10 bursting 50 chugging 50 pausing 40 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPi cell tonic)(format user)))

(run)

-39-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 45)(current-depth 71.7)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 90 tonic 70 bursting 30 chugging 10 pausing 10 fiber 10 multiple 23 artificial 10 injury 10 tremor 23 neg 24)(format dsp)))

(run)

-40-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 46)(current-depth 71.9)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPi cell none)(format user)))

(run)

-41-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 47)(current-depth 72.7)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer La cell border)(format user)))

(run)

-42-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 48)(current-depth 74)(dsp-cell sn_ratio 10 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10

pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer La cell none)(format user)))

(run)

-43-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 49)(current-depth 75.5)(dsp-cell sn_ratio 10 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer OT cell none)(format user)))

(run)

-44-

(assert (goal answer status available))

(assert (answer (timetag 50)(status endtrack)))

(run)

-45-

(assert (goal answer status available))

(assert (answer (timetag 51)(format message)))

(run)

-46-

(assert (goal answer status available))

(assert (answer (timetag 52)(status endop)))

(run)

Patient 2

```
;       ---------------------------------------------------------------------------------------------
;                               TEST CASE 2 : INPUT
;       Author: Linda Harley
;       Date:   October 28, 2003
;       File:   test2input.txt
;       Updates:
;       ---------------------------------------------------------------------------------------------
-1-
(assert (goal answer status available))
(assert (answer (timetag 1)(format message)(status received)))
(run)
-2-
(assert (goal answer status available))
(assert (answer (timetag 2)(text-field name "Patient 2" date "10/28/2003" doctor "Dr. X"
brainside "left")(format user)))
(run)
-3-
(assert (goal answer status available))
(assert (answer (timetag 3)(pathplan ST 50.35 Le 61.22 GPe 65.12 Li 70.18 GPi 71.56
La 75.72 OT 1000 IC 1000)(format model)))
(run)
```

-4-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 4)(current-depth 56.9)(dsp-cell sn_ratio 20 background 5 caudate 40 popcorn 60 HFD-P 20 LFD-B 20 border 20 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer ST cell none)(format user)))

(run)

-5-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 5)(current-depth 57)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 20 border 20 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer ST cell none)(format user)))

(run)

-6-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 6)(current-depth 60.22)(format depth)))

(run)

-7-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 7)(current-depth 63.17)(format depth)))

(run)

-8-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 8)(current-depth 65.12)(format depth)))

(run)

-9-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 9)(current-depth 66)(dsp-cell sn_ratio 5 background 5 caudate 10 popcorn 10 HFD-P 30 LFD-B 10 border 40 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 40 tremor 10 neg 10)(user layer GPe cell border)(format user)))

(run)

-10-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 10)(current-depth 67.3)(dsp-cell sn_ratio 10 background 10 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPe cell border)(format user)))

(run)

-11-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 11)(current-depth 67.6)(dsp-cell sn_ratio 20 background 10 caudate 10 popcorn 10 HFD-P 40 LFD-B 60 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPe cell HFD-P)(format user)))

(run)

-12-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 12)(current-depth 67.9)(dsp-cell sn_ratio 20 background 10 caudate 10 popcorn 10 HFD-P 80 LFD-B 30 border 10 tonic 10 bursting 10 chugging 20 pausing 20 fiber 20 multiple 20 artificial 10 injury 10 tremor 10 neg 10)(user layer GPe cell HFD-P)(format user)))

(run)

-13-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 13)(current-depth 68.4)(dsp-cell sn_ratio 20 background 10 caudate 10 popcorn 10 HFD-P 80 LFD-B 40 border 10 tonic 10 bursting 10 chugging 20 pausing 20 fiber 20 multiple 20 artificial 10 injury 10 tremor 10 neg 10)(user layer GPe cell HFD-P)(format user)))

(run)

-14-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 14)(current-depth 69)(dsp-cell sn_ratio 20 background 10 caudate 10 popcorn 10 HFD-P 60 LFD-B 80 border 10 tonic 10 bursting 70 chugging 20 pausing 70 fiber 20 multiple 20 artificial 10 injury 10 tremor 10 neg 10)(user layer GPe cell pausing)(format user)))

(run)

(assert (goal answer status available))

(assert (answer (timetag 15)(multichoice pausing)(format MC)))

(run)

-15-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 16)(current-depth 69.4)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 70 LFD-B 50 border 40 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(format dsp)))

(run)

-16-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 17)(current-depth 69.81)(format depth)))

(run)

-17-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 18)(current-depth 70.5)(dsp-cell sn_ratio 5 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 50 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(format dsp)))

(run)

-18-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 19)(current-depth 70.87)(format depth)))

(run)

-19-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 20)(current-depth 71.56)(format depth)))

(run)

-20-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 21)(current-depth 71.7)(dsp-cell sn_ratio 5 background 5 caudate 10 popcorn 10 HFD-P 50 LFD-B 10 border 90 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer Li cell border)(format user)))

(run)

-21-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 22)(current-depth 70.5)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 30 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPe cell LFD-B)(format user)))

(run)

-22-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 23)(current-depth 71.8)(dsp-cell sn_ratio 5 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 50 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer Li cell border)(format user)))

(run)

-23-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 24)(current-depth 71.9)(dsp-cell sn_ratio 5 background 5 caudate 10 popcorn 10 HFD-P 30 LFD-B 10 border 60 tonic 70 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer Li cell border)(format user)))

(run)

-24-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 25)(current-depth 72.8)(format depth)))

(run)

-25-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 26)(current-depth 72.8)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 30 border 10 tonic 10 bursting 10 chugging 10 pausing 80 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPi cell pausing)(format user)))

(run)

-26-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 27)(current-depth 73.1)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer GPi cell none)(format user)))

(run)

-27-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 28)(current-depth 73.65)(format depth)))

(run)

-28-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 29)(current-depth 74.3)(dsp-cell sn_ratio 20 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 30 chugging 30 pausing 10 fiber 10 multiple 10 artificial 10 injury 40 tremor 10 neg 10)(user layer GPi cell injury)(format user)))

(run)

(assert (goal answer status available))

(assert (answer (timetag 30)(multichoice injury)(format MC)))

(run)

-29-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 31)(current-depth 75.1)(dsp-cell sn_ratio 3 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 30 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer La cell none)(format user)))

(run)

-30-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 32)(current-depth 81)(dsp-cell sn_ratio 5 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer none cell none)(format user)))

(run)

-31-

(assert (goal name kbs status wakeup))

(assert (answer (timetag 33)(current-depth 80.4)(dsp-cell sn_ratio 5 background 5 caudate 10 popcorn 10 HFD-P 10 LFD-B 10 border 10 tonic 10 bursting 10 chugging 10 pausing 10 fiber 10 multiple 10 artificial 10 injury 10 tremor 10 neg 10)(user layer none cell none)(format user)))

(run)

-32-

(assert (goal answer status available))

(assert (answer (timetag 34)(status endtrack)))

(run)

-33-

(assert (goal answer status available))

(assert (answer (timetag 35)(format message)))

-34-

(assert (goal answer status available))

(assert (answer (timetag 36)(status endop)))

(run)

G.3. Test Results

Patient 1

```
;       ---------------------------------------------------------------------------------------------
;                               TEST CASE 1 : ACTUAL OUTPUT
;       Author: Linda Harley
;       Date:   October 28, 2003
;       File:   test2outputactual.txt
;       Updates:
;       ---------------------------------------------------------------------------------------------
-1-
M:message*("")*("The KBS has been activated and is standing by.")!
-2-
R:user*("name,date,doctor,brainside")*(nil)!
-3-
R:model*(first pathplan)*(nil)!
-4-
D:wakeup*(52.85)*(nil)!
D:data*(Layer ST Cell popcorn State solid)*(nil)!
-5-
D:data*(Layer ST Cell popcorn State done)*(nil)!
-6-
D:wakeup*(55.345)*(nil)!
D:data*(Layer ST Cell none State done)*(nil)!
```

-7-

D:wakeup*(56.84)*(nil)!

D:data*(Layer Le Cell none State fuzz)*(nil)!

-8-

D:data*(Layer ST Cell none State done)*(nil)!

-9-

D:wakeup*(60.225)*(nil)!

D:data*(Layer GPe Cell none State fuzz)*(nil)!

-10-

D:data*(Layer GPe Cell none State done)*(nil)!

-11-

Q:MC*(popcorn HFD-P LFD-B border tonic bursting chugging pausing fiber multiple artificial injury tremor)*(users celltype selection conflicts with pathplan and dsp analysis. Please reselect celltype.)!

D:data*(Layer ST Cell popcorn State done)*(nil)!

-12-

D:wakeup*(62.61)*(nil)!

D:data*(Layer GPe Cell none State done)*(nil)!

-13-

Q:MC*(popcorn HFD-P LFD-B border tonic bursting chugging pausing fiber multiple artificial injury tremor)*(Users celltype selection conflicts with pathplan and dsp analysis. Please reselect celltype.)!

D:data*(Layer ST Cell popcorn State done)*(nil)!

-14-

D:data*(Layer Le Cel border State done)*(nil)!

-15-

D:data*(Layer GPe Cell HFD-P STate done)*(nil)!

-16-

D:data*(Layer none Cell none State none)*(nil)!

-17-

D:wakeup*(64.02)*(nil)!

D:data*(Layer GPe Cell none State done)*(nil)!

-18-

D:data*(Layer GPe Cell HFD-P State done)*(nil)!

-19-

D:data*(Layer GPe Cell LFD-B State done)*(nil)!

-20-

D:data*(Layer none Cell none State none)*(nil)!

-21-

Q:MC*(popcorn HFD-P LFD-B border tonic bursting chugging pausing fibermultiple artificial injury tremor)*(User celltype selection conflicts with pathplan and dsp analysis. Please reselect celltype.)!

D:data*(Layer GPe Cell HFD-P State done)*(nil)!

-22-

D:wakeup*(64.43)*(nil)!

D:data*(Layer Li Cell none State fuzz)*(nil)!

-23-

D:data*(Layer GPe Cell border State done)*(nil)!

-24-

D:wakeup*(66.32)*(nil)!

D:data*(Layer GPi Cell none State fuzz)*(nil)!

-25-

D:data*(Layer GPe Cell HFD-P State done)*(nil)!

-26-

Q:MC*(popcorn HFD-P LFD-B border tonic bursting chugging pausing fibermultiple artificial injury tremor)*(User celltype selection conflicts with pathplan and dsp analysis. Please reselect celltype.)!

D:data*(Layer GPe Cell HFD-P State done)*(nil)!

-27-

D:data*(Layer Li Cell border State done)*(nil)!

-28-

D:wakeup*(67.21)*(nil)!

D:data*(Layer GPi Cell none State done)*(nil)!

-29-

D:wakeup*(68.21)*(nil)!

D:data*(Layer Li Cell border State done)*(nil)!

-30-

D:wakeup*(68.21)*(nil)!

D:data*(Layer GPi Cell tonic State done)*(nil)!

-31-

M:message*nil*Warning: Proble close to the Optic Track and Internal Capsule. Proceed with caution.!

D:wakeup*(68.21)*(nil)!

D:data*(Layer La Cell none State fuzz)*(nil)!

-32-

D:data*(Layer GPi Cell tonic State done)*(nil)!

-33-

D:data*(Layer GPi Cell pausing State done)*(nil)!

-34-

D:data*(Layer GPi Cell pausing State done)*(nil)!

-35-

D:data*(Layer GPi Cell bursting State done)*(nil)!

-36- NOTE: JUST REMOVE ANY " FROM THE STRING.

Q:MC*(Striatum Le GPe Li GPi La Optic InternalCapsule)*("According to KBS the probe is in layer" La "User is in conflict. Please re-select layer type.")!

D:data*(Layer GPi Cell border State done)*(nil)!

-37-

D:data*(Layer GPi Cell chugging State done)*(nil)!

-38-

D:data*(Layer GPi Cell tonic State done)*(nil)!

-39-

D:data*(Layer La Cell border State done)*(nil)!

-40-

D:data*(Layer GPi Cell none State done)*(nil)!

-41-

D:data*(Layer La Cell border State done)*(nil)!

-42-

D:data*(Layer La Cell none State done)*(nil)!

-43-

D:data*(Layer OT Cell none State solid)*(nil)!

-44-

D:KBS Pathplan*ST 50.6 solid Le 55.345 fuzz GPe 56.84 fuzz Li 64.1 fuzz GPi 4.46 fuzz La 68.28 fuzz OT 75.5 solid IC 1000 nil*!

M:message*(nil)*("Move 2mm posterior.")!

-45-

nothing

-46-

nothing. NOTE: a kbsdata.txt file is created.

Patient 2

```
;       ---------------------------------------------------------------------------------------------
;                               TEST CASE 2 : ACTUAL OUTPUT
;       Author: Linda Harley
;       Date:   October 28, 2003
;       File:   test2outputactual.txt
;       Updates:
;       ---------------------------------------------------------------------------------------------
-1-
M:message*("")*("The KBS has been activated and is standing by.")!
-2-
R:user*("name,date,doctor,brainside")*(nil)!
-3-
R:model*(first pathplan)*(nil)!
-4-
D:wakeup*(60.22)*(nil)!
D:data*(Layer ST Cell popcorn State solid)*(nil)!

-5-
D:data*(Layer ST Cell none State done)*(nil)!
-6-
D:wakeup*(63.17)*(nil)!
D:data*(Layer ST Cell none State done)*(nil)!
```

-7-

D:wakeup*(65.12)*(nil)!

D:data*(Layer Le Cell none State fuzz)*(nil)!

-8-

D:wakeup*(67.65)*(nil)!

D:data*(Layer GPe Cell none State fuzz)*(nil)!

-9-

D:data*(Layer GPe Cell border State done)*(nil)!

-10-

D:data*(Layer GPe Cell border State done)*(nil)!

-11-

D:data*(Layer GPe Cell HFD-P State done)*(nil)!

-12-

D:wakeup*(69.18)*(nil)!

D:data*(Layer GPe Cell HFD-P State done)*(nil)!

-13-

D:data*(Layer GPe Cell HFD-P State done)*(nil)!

-14-

Q:MC*(popcorn HFD-P LFD-B border tonic bursting chugging pausing fiber multiple artificial injury tremor)*(Users cell type selection conflicts with pathplan and dsp analysis. Please reselect celltype.)!

D:data*(Layer GPe Cell pausing State done)*(nil)!

-15-

D:wakeup*(70.87)*(nil)!

D:data*(Layer GPe Cell HFD-P State done)*(nil)!

-16-

D:data*(Layer GPe Cell none State done)*(nil)!

-17-

D:data*(Layer none Cell none State none)*(nil)!

-18-

D:wakeup*(71.56)*(nil)!

D:data*(Layer Li Cell none State fuzz)*(nil)!

-19-

D:wakeup*(73.64)*(nil)!

D:data*(Layer GPi Cell none State fuzz)*(nil)!

-20-

D:data*(Layer Li Cell border State done)*(nil)!

-21-

D:data*(Layer GPe Cell LFD-B State done)*(nil)!

-22-

D:data*(Layer Li Cell border State done)*(nil)!

-23-

D:data*(Layer Li Cell border State done)*(nil)!

-24-

D:data*(Layer GPi Cell none State done)*(nil)!

-25-

D:data*(Layer GPi Cell pausing State done)*(nil)!

-26-

D:data*(Layer GPi Cell none State done)*(nil)!

-27-

D:wakeup*(74.72)*(nil)!

D:data*(Layer GPi Cell none State done)*(nil)!

-28-

Q:MC*(popcorn HFD-P LFD-B border tonic bursting chugging pausing fiber multiple artificial injury tremor)*(Users celtype selection conflicts with pathplan and dsp analysis. Please reselect celltype.)!

D:data*(Layer GPi Cell injury State done)*(nil)!

-29-

D:wakeup*(75.72)*(nil)!

D:data*(Layer La Cell none State solid)*(nil)!

-30-

D:wakeup*(75.72)*(nil)!

M:message*(nil)*(Warning: Probe close to Opctic Track and Internal Capsule. Proceed with catuion.")!

D:data*(Layer none Cell none State none)*(nil)!

-31-

D:data*(Layer none Cell none State done)*(nil)!

-32-

D:KBS Pathplan*ST 56.9 solid Le 63.17 fuzz GPe 65.12 fuzz Li 70.87 fuzz GPi 71.56 fuzz La 75.1 solid OT 1000 nil IC 1000 nil*!

M:message*(nil)*("Probe is posterior. Move 1mm posterior to find Internal Capsule.")!

-33-

nothing

nothing

REFERENCES

[1] *Basic Information About Parkinson's Disease,* [Online], Staten Island, NY, American Parkinson Disease Association 2003, Available: http://www.apdaparkinson.org/user/AboutParkinson.asp [Accessed 20 September 2003].

[2] Iacono R. P., *Learn About Parkinson's Disease* [Online], Neuroscience Clinic, Available: http://www.pallidotomy.com/pathology.html [Accessed 17 September 2003].

[3] Iacono R. P., *Learn About Neurosurgical Procedures* [Online], Neuroscience Clinic, Available: http://www.pallidotomy.com/pallidotomy.html [Accessed 17 September 2003].

[4] Ehringer H., Hornykiewicz O. *Verteilung von Noradrenalin und Dopamin (3-Hydroxytyramin) im Gehirn des Menschen und ihr Verhalten bei Erkrankungen des Extrapyramidalen Systems*. Klin Wochenschr, 1960, pp. 1236-1239.

[5] Delwaide, P. J. and Gonce M., *Pathophysiology of Parkinson's Signs*. Jankovic, J. and Tolosa, E. (eds.) Parkinson's Disease and Movement Disorders. 2nd ed. Baltimore, Maryland: Williams & Wilkins, 1993, pp. 77-92.

[6] Shahani B. T., Young R. R., *Physiological and Pharmacological Aids in the Differential Diagnosis of Tremor.* Journal Neural Neurosurgery Psychiatry, 1976, pp. 772-783.

[7] Cotzias C. G., Van Woert M. H., Schiffer L. M., *Aromatic Amino Acids and Modification of Parkinsonism.* N Engl J Med, 1967, pp. 374-379.

[8] Poewe W., *L-dopa in Parkinson's Disease: Mechanisms of Action and Pathophysiology of Late Failure*. Jankovic, J. and Tolosa, E. (eds.) Parkinson's Disease and Movement Disorders. 2nd ed. Baltimore, Maryland: Williams & Wilkins, 1993, pp. 103-114.

[9] Günewald R., *Treatment of Parkinsons Disease.* [Online] England, Available: http://www.shef.ac.uk/~smtw/2000/ns/ns0428b.htm [Accessed 19 September 2003].

[10] Grossman R., and Hamilton W. J., *Surgery for Movement Disorders*, Jankovic, J. and Tolosa, E. (eds.) Parkinson's Disease and Movement Disorders. 2nd ed. Baltimore: Williams & Wilkins, 1993, pp. 531-548.

[11] Medtroni, Inc., *Activa System,* [Online], USA, Available: http://www.medtronic.com/activa/physician/implantable.html [Accessed 30 September 2003].

[12] Schaltenbrand G. and Bailey P. *Introduction to Stereotaxi with an Atlas of the Human Brain.* Stuttgart: Thieme 1959.

[13] *On-Line Medial Dictionary* [Online], University of Newcastel upon Tyne, Department of Medical Oncology, Available: http://cancerweb.ncl.ac.uk/omd/index.html [Accessed 7 August 2003].

[14] Vitek, J. L., Bakay, R. A., DeLong, M. R., *"Microelectrode-Guided Pallidotomy for Medically Intractable Parkinson's Disease." Advanced Neurology* 1997, pp. 183-198.

[15] The Virginia Spine Institute, *Stealthstation®*, [Online], USA, Available: http://www.spinemd.com/2-stealth.htm [Accessed 30 September 2003].

[16] FHC, *Axon Instruments Guideline System 3000A,* [Online], USA, Available: http://www.fy-co.com/axong_gs3000a.html [Accessed 30 September 2003].

[17] Gibson, V., Peifer, J., Gandy M., Robertson, S. and Mewes, K. (2003). *3D Visualization Methods to Guide Surgery for Parkinson's Disease*. Proceedings of the 11th Annual Medicine Meets Virtual Reality (MMVR) Conference, Newport Beach, CA, January 2003, p. 86-92.

[18] Kochan, S.G. *Programming in ANSI C*. Revised ed. Indianapolis: SAMS, 1994. pp. 1-22.

[19] Chorniak, E. and McDermott D. *Introduction to Artificial Intelligence*. Reading: Addison-Wesley Publishing Company, 1985. pp 1-8.

[20] Haykin, S. *Neural Networks a Comprehensive Foundation.* Englewood Cliffs: Macmillan College Publishing Company, Inc., 1994. pp1-41.

[21] Zadeh, L.A., *Knowledge Representation in Fuzzy Logic* IEEE Transactions on Knowledge and Data Engineering, Vol. 1., No.1., March 1989. pp 89-99.

[22] Velasevic, D. M., Saletic, D. Z., Saletic, S. Z., *A Fuzzy Sets Theory Application in Determining the Severity of Respiratory Failure.* International Journal of Medical Informatics 63 (2001) pp 101-107. [Online] Available: www.elsevier.com/locate/ijmedinf. [Accessed: 30 September 2003].

[23] Von Altrock, C. *Fuzzy Logic & NeuroFuzzy Applications Explained*. Englewood Cliffs: Prentice Hall PTR, 1995. pp 1-28.

[24] Schwartz T., *Fuzzy Systems in the Real World*, AI Expert, 5(1990) pp 28-36.

[25] Schimdt Rainer, Montani Stefania, Bellazzi Riccardo, Portinale Luigi and Gierl Lothar. *Case-Based Reasoning for Medical Knowledge-Based Systems.* International Journal of Medical Informatics 64 (2001) pp355-367. [Online] Available: www.elsevier.com/locate/ijmedinf [Accessed: 30 September 2003].

[26] Kolodner, J. *Case-Based Reasoning.* San Mateo: Morgan Kaufmann Publishers, Inc., 1993. pp 1-10.

[27] Giarratano, J. and Riley, G. *Expert System Principles and Programming.* 2nd edition. Boston:PWS, 1994, pp.1-11.

[28] Ericsson, K.A., and Simon, H.A. *Protocol Analysis: Verbal Reports as Data.* Revised Edition. Cambridge: The MIT Press, 1993.

[29] *CLIPS: A Tool for Building Expert Systems.* [Online], GHG Internet Services, Available: http://www.ghg.net/clips/CLIPS.html [Accessed: 13 August 2003].

[30] Mark Tomlinson, May 18 2003, *CLIPS ActiveX Control* [Online], Available: http://www.tomlinson-web.net, [Accessed 13 August 2003].

[31] Freudenrich, C.C., HowStuffWorks, Inc. (No date) *How Oil Drilling Works.* [Online] Atlanta Gerogia, Available: http://science.howstuffworks.com/oil-drilling1.htm [Accessed: 20 December 2003.]

[32] U.S. Department of the Interior and U.S. Geological Survey. 7 April 2003. *The Modified Mercalli Intensity Scale.* [Online]. Denver, U.S. Geological Survey General Interest Publication. U.S. Government printing office: 1989-288-913,

Available: http://neic.usgs.gov/neis/general/mercalli.html [Accessed: 20 November 2003.]